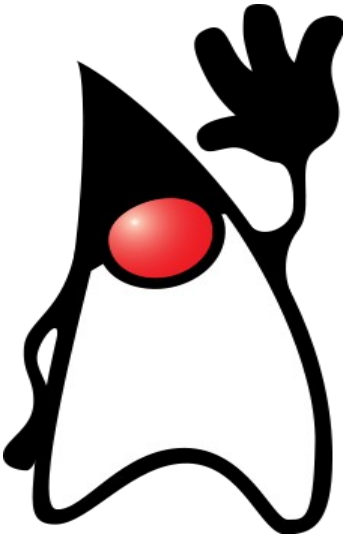
ΑΛΕΞΑΝΔΡΕΙΟ Τ.Ε.Ι. ΘΕΣΣΑΛΟΝΙΚΗΣ
ΣΧΟΛΗ ΤΕΧΝΟΛΟΓΙΚΩΝ ΕΦΑΡΜΟΓΩΝ
ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ

Πτυχιακή εργασία

ΕΦΑΡΜΟΓΗ J2ME ΓΙΑ ΚΙΝΗΤΟ ΜΕ ΧΡΗΣΗ NETBEANS



Του Φοιτητού
Κωνσταντίνου Χρόνη
ΑΜ: 04/2564

Επιβλέπων Καθηγητής
Παναγιώτης Σφέτσος

Ευχαριστίες κι αφιέρωση

Με την ολοκλήρωση αυτής της εργασίας, σηματοδοτείται η απόκτηση του πτυχίου μου και το ξεκίνημα μίας καινούριας αρχής στην ζωή μου από εδώ και στο εξής. Όλα αυτά όμως δεν θα ήταν τίποτα χωρίς την στήριξη κάποιων ανθρώπων.

Η εργασία αυτή, αφιερώνεται στους γονείς μου που με τις προσπάθειές τους όλα αυτά τα χρόνια με βοήθησαν να σπουδάσω και να γίνω ο άνθρωπος που είμαι.

Αφιερώνεται επίσης και σε ένα άτομο το οποίο από την στιγμή που το γνώρισα με στήριζε κάθε στιγμή, σε όποια δυσκολία κι αν αντιμετώπισα. Δυστυχώς, δεν πρόλαβε να δει την ολοκλήρωση αυτού το έργου και το τέλος της φοιτητικής μου ζωής. Είμαι σίγουρος πως όπου κι αν βρίσκεται, θα νιώθει περήφανη.

Ευχαριστώ όλους όσους με έχουν στηρίξει όλα αυτά τα χρόνια. Να 'στε όλοι πάντα καλά!

Περιεχόμενα

Ευρετήριο Πινάκων

Ευρετήριο Εικόνων

Περίληψη

Αυτή η πτυχιακή εργασία έχει ως αρχικό στόχο, να κάνει μία μικρή παρουσίαση της Java Micro Edition και να εισάγει τον αναγνώστη στον μικρό, αλλά με μεγάλες δυνατότητες κόσμο της. Ο άλλος της στόχος είναι η δημιουργία μία εφαρμογής, σχετικά με τις Επαφές, τις Σημειώσεις και τις Υποχρεώσεις, η χρήση των οποίων είναι μία από τις πιο κοινές ενασχολήσεις οποιουδήποτε έχει μία κινητή συσκευή και η πιθανή απώλειά τους, θα δημιουργούσε αρκετά μεγάλο πρόβλημα. Διαβάζοντας τις παρακάτω σελίδες, ο αναγνώστης θα μάθει πράγματα όσον αφορά την εξέλιξη της γλώσσας, καθώς και τις ομάδες προτύπων και συστατικών μερών τα οποία την αποτελούν. Τα παραπάνω συνιστούν το κύριο μέρος αυτού του πονήματος, ενώ ακόμη γίνεται και αναφορά στα προβλήματα τα οποία αποτελούν εμπόδιο για την καθολική αποδοχή της γλώσσας αυτής, από τους κατασκευαστές κινητών συσκευών, καθώς και στα διάφορα εργαλεία για την ανάπτυξή της. Με την ολοκλήρωση της ανάγνωσής της, ο αναγνώστης όχι μόνο θα έχει κατανοήσει βασικά στοιχεία της Java Micro Edition, αλλά και τον τρόπο για την ανάπτυξη μίας εφαρμογής, ανάλογα με τις ανάγκες του κάθενός.

Abstract

This thesis original objective is to make a short presentation of the Java Micro Edition and introduces the reader to a small world, but of great potential. The other objective is to create an application on Contacts, Events and ToDos, the use of which is one of the most common occupations has any one on a mobile device and the possible loss, would create a big problem. By reading the following pages, readers will learn things about the evolution of language and groups of standards and components that make up that. These constitute the main part of this thesis, and even makes reference to problems which constitute the major obstacle to the universal acceptance of this language from the manufacturers of mobile devices, and to various tools for development. Upon completion of the reading, the reader will not only understand basic elements of Java Micro Edition, but also how to develop an application, depending on individual needs.

Κεφάλαιο 1ο

1.1 Εισαγωγή

Σε αυτό το κεφάλαιο θα δούμε ένα εισαγωγικό κείμενο, το οποίο προϊδεάζει τον αναγνώστη για ορισμένα πράγματα τα οποία θα δει αργότερα. Επίσης γίνεται μία σύγκριση των προγραμμάτων για κινητές συσκευές μερικών μεγάλων εταιρειών, ώστε να κατανοήσει ο αναγνώστης την σημασία ανάπτυξης της εφαρμογής.

1.2 Εισαγωγικό κείμενο

Τα τελευταία 15 χρόνια τα κινητά τηλέφωνα και μετέπειτα οι κινητές συσκευές, έχουν γίνει αναπόσπαστο κομμάτι του μέσου ανθρώπου.. Ένα κινητό τηλέφωνο από ένα απλό μέσο φωνητικής επικοινωνίας με κάποιον για λόγους ανάγκης, έχει γίνει πια ένα αναπόσπαστο κομμάτι της καθημερινής μας ζωής. Με μία κινητή συσκευή, μπορεί κάποιος να επικοινωνήσει γραπτά, με εικόνες, μπορεί να πλοηγηθεί στο Internet, να παίξει παιχνίδια, να ακούσει ραδιόφωνο, να ακούσει μουσική, να δει βίντεο, να τραβήξει φωτογραφίες και βίντεο, να την χρησιμοποιήσει στην δουλειά του και κυρίως αν και δεν γίνεται άμεσα αντιληπτό, να οργανώσει την ζωή του. Εκατοντάδες επαφές τηλεφώνων υπάρχουν σε κάθε κινητή συσκευή, χωρίς πιθανότατα αυτές να έχουν σωθεί κάπου αλλού ψηφιακά ή ακόμα και γραπτά. Ακόμη κάποιος αποθηκεύει σημειώσεις για την δουλειά, του, όπως κάποιες συναντήσεις οι οποίες είναι προγραμματισμένες, κοινωνικές υποχρεώσεις, όπως τα γενέθλεια κάποιου, αλλά και πολύ απλές υπενθυμίσεις όπως το να πληρώσει το ενοίκιο, ή να πάει στο σούπερ μάρκετ για να ψωνίσει. Έτσι σε πιθανή απώλεια της συσκευής, υπάρχει και πιθανή απώλεια όλων των παραπάνω πληροφοριών οι οποίες μπορεί να είναι από ασήμαντες έως ζωτικής σημασίας για κάποιους.

Για αυτό τον σκοπό έχουν αναπτυχθεί διάφορα προγράμματα από την κάθε εταιρεία κινητής συσκευής, ώστε να εξυπηρετήσει τις ανάγκες των πελατών της για την αποθήκευση κάθε είδους πληροφορίας η οποία μπορεί να υπάρχει μέσα στα προϊόντα της. Τα προγράμματα αυτά θα παρουσιαστούν παρακάτω και με μία μικρή ανάλυση θα γίνει φανερό πως η χρήση τους αλλά και η απόκτησή τους δεν είναι τόσο εύκολη όσο θα έπρεπε. Οι περιορισμοί αρκετοί και κάποιες φορές καταδικαστικοί για χρήστες υπολογιστών που δεν έχουν κάποια έκδοση του λειτουργικού συστήματος Windows, αλλά κάποια διανομή Linux.

Σκοπός αυτής της πτυχιακής εργασίας είναι η ανάπτυξη ενός προγράμματος αποθήκευσης κάποιων πληροφοριών, χωρίς ιδιαίτερες απαιτήσεις software και hardware. Μία λύση δωρεάν για κάθε χρήστη κινητής συσκευής και ηλεκτρονικού υπολογιστή. Η μόνη προϋπόθεση; Ένας υπολογιστής, ένας τρόπος σύνδεσης του υπολογιστή με την κινητή συσκευή και το .jar αρχείο αυτής της πτυχιακής εργασίας.

Μιας και το πρόγραμμα πρέπει να λειτουργεί παντού, επιλέχθηκε η γλώσσα προγραμματισμού Java και συγκεκριμένα η Micro Edition έκδοση. Μία έκδοση αρκετά περιορισμένη σε σχέση με αυτό που έχουν οι περισσότεροι προγραμματιστές για την γλώσσα αυτή. Τα τελευταία χρόνια γίνονται προσπάθειες βελτίωσής της, όμως η όχι και τόσο μεγάλη ανάπτυξη της σε δυνατότητες, αλλά κυρίως και οι διαφορετικές υλοποιήσεις της, από τον κάθε κατασκευαστή κινητής συσκευής, έχουν

δημιουργήσει αρκετά ζητήματα στην ανάπτυξη "σοβαρών" εφαρμογών. Με τον όρο σοβαρό αναφέρομαι στην μη δημιουργία παιχνιδιών, μιας και κυρίως αυτή είναι η χρήση της, λόγω των προβλημάτων που μόλις αναφέρθηκαν.

Η J2ME, όπως είναι εν συντομία το όνομά της, έχει όλες τις καλές δυνατότητες των πιο πλήρεις εκδόσεών της, αλλά και το "βάρος" ενός πούπουλου. Είναι μία πολύ "ελαφριά" γλώσσα, ώστε να μπορεί το κάθε παραγόμενο αποτέλεσμά της να τρέχει σε κινητές συσκευές με μικρές δυνατότητες και πολύ περιορισμένη μνήμη. Έχει όμως και τα μειονεκτήματα των γλωσσών οι οποίες εξαρτώνται από την πλατφόρμα. Μπορεί ένα J2ME πρόγραμμα να παίζει σε κάθε πλατφόρμα που υποστηρίζει Java, όμως η εκτέλεση της εφαρμογής διαφέρει από συσκευή σε συσκευή. Αυτό δεν είναι ένα πρόβλημα της γλώσσας αυτής καθεαυτής, αλλά των κατασκευαστριών εταιρειών κινητών συσκευών οι οποίες σχεδιάζουν την κάθε συσκευή σύμφωνα με τα δικά τους πρότυπα, αδιαφορώντας για τις άλλες εταιρείες, δημιουργόντας το φαινόμενο του κατακερματισμού, το οποίο είναι το κύριο μειονέκτημα της Java Micro Edition.

Ένα παιχνίδι παραδείγματος χάρην, μπορεί να λειτουργεί κανονικά σε μία συσκευή Nokia κι όχι σε μία συσκευή Sony Ericsson. Το χειρότερο όμως, μπορεί να τρέχει σε ένα μοντέλο μιας εταιρείας και να μην τρέχει σε άλλο μοντέλο της ίδίας, ακόμα κι αν βγήκαν τον ίδιο καιρό στην αγορά. Αυτό το "πονοκεφάλιασμα" στον χρήστη, είναι κάτι το οποίο οι δημιουργοί της J2ME προσπαθούν να επιλύσουν, βγάζοντας κάποια πρότυπα και κάποιες "ομπρέλες" προτύπων, με το σκεπτικό πως όσοι κατασκευαστές τις υιοθετούν, τότε η μεταφερσιμότητα μίας εφαρμογής, δεν θα είναι πρόβλημα, όπως και είναι άλλωστε η φιλοσοφία και ο τρόπος λειτουργίας της Java.

Στην προσπάθεια αντιμετώπιση αυτού του προβλήματος, οι εταιρείες οι οποίες προσφέρουν προγράμματα όπως τα IDE για την ανάπτυξη εφαρμογών σε J2ME, έχουν φτιάξει κάποιους προσομοιωτές, τα Emulators, τα οποία αναπαριστούν κάποιο τύπο ή ακόμα και κάποιο μοντέλο κινητού για την όσο το δυνατό πιο ρεαλιστικό αποτέλεσμα. Το NetBeans, το Eclipse, η Sun, ακόμα και η ίδια η Nokia έχουν τέτοιους Emulators.

Η Java Micro Edition έχει διεισδύσει για τα καλά στην ζωή μας, ακόμα κι αν δεν το έχουμε καταλάβει, μιας και όλοι χρησιμοποιούμε μία κινητή συσκευή με κάποια εφαρμογή μέσα της. Σαν αποτέλεσμα, πολλοί είναι εκείνοι που θέλουν να φτιάξουν μία εφαρμογή για τον εαυτό τους και τους άλλους, είτε για να κερδίσουν χρήματα, είτε για να βοηθήσουν ή ακόμα και για να δημιουργήσουν. Για να είναι εφικτό κάτι τέτοιο, πρέπει να δουν έως ένα βαθμό την γλώσσα και να κατανοήσουν την λιτότητά της, αλλά και τα μεγάλα αποτελέσματα τα οποία μπορεί να παράξει.

Η J2ME βασίζεται σε ένα Configuration, όπως τα CDC και CLDC, σε ένα Profile, όπως το MIDP και σε πάρα πολλά Optional Packages. Χωρίς τουλάχιστον τα δύο πρώτα συστατικά μέρη, δε γίνεται να φτιαχτεί κάποια εφαρμογή, ενώ χωρίς το τρίτο, συνήθως θα είναι χωρίς ιδιαίτερη χρησιμότητα. Το πλήθος των τριών αυτών συστατικών μερών, αυξάνεται όσο περνούν τα χρόνια με ιδιαίτερη έμφαση στα Optional Packages τα οποία φτάνουν σε αριθμό κοντά τα 300.

Μία εφαρμογή μπορεί να επικντρωθεί σε πάρα πολλούς τομείς όπως το Bluetooth, τα πολυμέσα, τα προσωπικά δεδομένα, τα SMS – MMS, η αποθήκευση δεδομένων σε μία Βάση Δεδομένων, η πλοήγηση στο Internet, οι τραπεζικές συνδιαλλαγές και τα πολύ γνωστά σε όλους Java games. Όσο πιο

πολλά Optional Packages δημιουργούνται, τόσες πιο πολλές και οι δυνατότητες της.

Και δεν είναι μόνο τα παραπάνω τα οποία μπορεί κάποιος να καταφέρει με την J2ME. Καθημερινά σχεδόν καινούριες εφαρμογές κυκλοφορούν στο Internet οι οποίες βασίζονται σε προσωπικές βιβλιοθήκες.

Τον τελευταίο καιρό όμως έχει επέλθει η επανάσταση. Το iPhone και το HTC έχουν κατακλίσει την αγορά ορίζοντας εκ νέου το τι μπορεί να κάνει μία κινητή συσκευή. Όλο και περισσότεροι άνθρωποι σπεύδουν να γευτούν τους καρπούς της τεχνολογίας, οι οποίοι καθημερινά αυξάνονται, ειδικά με τις εφαρμογές τις οποίες γράφουν οι ίδιοι οι χρήστες, όσον αφορά την πρώτη συσκευή.

Η ανάλυση των παραπάνω κινητών συσκευών, ξεφεύγει από τα όρια αυτής της εργασίας, όμως διαβάζοντας τα κεφάλαια αυτού του πονήματος, ο αναγνώστης θα έχει την δυνατότητα να κατανοήσει την δομή της J2ME, τις δυνατότητές της, τον τρόπο ανάπτυξης εφαρμογών σε αυτή και εν κατακλείδει, θα έχει στην διάθεση του μία εφαρμογή αρκετά χρήσιμη για την κινητή του συσκευή, η οποία θα μπορέσει να τον γλυτώσει από αρκετά προβλήματα.

1.3 Λόγοι ανάπτυξης της εργασίας

Ο κύριος λόγος ανάπτυξης αυτής της πτυχιακής, είναι η δημιουργία ενός προγράμματος, το οποίο θα είναι ελεύθερο προς όλους τους χρήστες κινητών τηλεφώνων, οι οποίοι θα μπορούν να το χρησιμοποιήσουν για όποια συσκευή κι αν έχουν, χωρίς την ανάγκη υπολογιστή, ο οποίος θα έχει συγκεκριμένο λειτουργικό σύστημα, σε συγκεκριμένη έκδοση κάνοντας την χρήση κάποιου προγράμματος εξαναγκαστικά, μη έχοντας άλλη λύση. Πρόγραμματα τα οποία έχουν πρόσβαση σε δεδομένα κινητών, έχουν όλες οι εταιρείες. Μερικά είναι πιο εξελιγμένα κι άλλα λιγότερο. Το σίγουρο πάντως είναι πως οι απαιτήσεις τους είναι σχετικά υψηλές ως προς το λειτουργικό σύστημα, και σαν αποτέλεσμα και ως προς το μηχάνημα. Ας δούμε όμως μερικά από τα προγράμματα των πιο γνωστών εταιρειών στον χώρο των κινητών συσκευών.

1.4 Προγράμματα διαφόρων εταιρειών

1.4.1 Nokia - PC Nokia Suite

Η Nokia τα τελευταία χρόνια έχει αναπτύξει το PC Nokia Suite με σκοπό να καλύψει τις ανάγκες των πελατών της. Το πρόγραμμά της είναι αρκετά πλήρες και για αυτό οι απαιτήσεις του είναι σχετικά υψηλές.

Οι απαιτήσεις του Nokia PC Suite είναι οι εξής:

Windows Vista 32-bit και 64-bit εκδόσεις
Windows XP Professional x64 Edition (Service Pack 2)
Windows XP (Professional ή Home Edition) Service Pack 2 ή επόμενο
Σκληρός δίσκος: τουλάχιστον 300 MB ελεύθερα

Το μέγεθος του εκτελέσιμου αρχείου ανέρχεται στα 32.2 MB, ένα μέγεθος αρκετά μεγάλο για μία απλή

σύνδεση στο Internet [14].

1.4.2 Sony Ericsson - Sony Ericsson PC Suite

Η Sony Ericsson έχει ένα δικό της πρόγραμμα, το οποίο υστερεί σε αρκετά θέματα σε σχέση με το Nokia PC Suite. Είναι όμως η μόνη εταιρεία που τουλάχιστον παρέχει κάποια plugins για άλλο λειτορυγικό σύστημα, εκτός από Windows.

Οι απαιτήσεις του Sony Ericsson PC Suite είναι οι εξής:

- Pentium II – 233 MHz ή παραπάνω
- Windows XP: 128 MB RAM
- Windows Vista: 512 MB RAM
- 50 MB σκληρός δίσκος
- Windows XP Home, Pro, Media Center (SP1/SP2/SP3)
- Windows Vista 32 και 64 bits Ultimate, Enterprise, Business, Home Premium and Home Basic (Με ή χωρίς SP1)

Το εκτελέσιμο αρχείο έχει μέγεθος 15.5 MB [26].

1.4.3 Sony Ericsson iSync plugins for Mac OS X

Το πρόγραμμα αυτό είναι μόνο για Apple υπολογιστές με σκοπό των συγχρονισμό των PIM συστατικών μερών και μόνο, εν αντιθέσει με το Sony Ericsson PC Suite το οποίο έχει αρκετές παραπάνω λειτουργίες.

Οι απαιτήσεις του είναι:

- Tiger: Mac OS 10.4.9 - 10.4.11 (iSync 2.4 έκδοση)
- Leopard: Mac OS 10.5.1 - 10.5.6 (iSync 3.0 - 3.0.2 έκδοση)

Το εκτελέσιμο αρχείο έχει μέγεθος 7.8 MB [27].

1.4.4 Motorola - Motorola Phone Tools, Motorola Mobile Phone USB Drivers

Η Motorola τα τελευταία χρόνια δεν φαίνεται και πολύ στον χώρο της κινητής τηλεφωνίας εν συγκρίσει με την Nokia και την Sony Ericsson. Ίσως και για αυτό το λόγο να χρεώνει το πρόγραμμα της, με το οποίο ο χρήστης μπορεί να έχει πρόσβαση στο κινητό για αυξημένη λειτουργικότητα με αυτό.

Motorola Phone Tools

Η τιμή του ανέρχεται στα $49.99 ενώ οι απαιτήσεις του είναι:

- Windows® 2000, XP, ή Vista.
- Pentium® II 233 MHz
- 64MB RAM για Windows® 2000 και XP, 128MB RAM για Vista
- 120 MB σκληρού δίσκου
- Internet Explorer 6.0 ή επόμενη έκδοση

[12]

Motorola Mobile Phone USB Drivers

Ακόμη υπάρχει και αυτό το πρόγραμμα, για μία απλή ανάγνωση των επαφών από το κινητό.

Windows 2000® 32 bit ή 64 bit
Windows XP® 32 bit ή 64 bit
Windows Vista® 32 bit ή 64 bit

Το συγκεκριμένο λογισμικό εκτός του ότι είναι μόνο για τα παραπάνω λειτουγικά συστήματα, χρειάζεται και ειδικούς USB drivers για να μπορέσει να συνδεθεί το κινητό με τον υπολογιστή [13].

1.4.5 LG - LG Outlook Sync

Το LG Outlook Sync είναι ένα πρόγραμμα συγχρονισμού μίας LG συσκευής με το Outlook, με σκοπό την μεταφορά επαφών, σημειώσεων και υποχρεώσεων. Όμως η εφαρμογή αυτή υποστηρίζει μόνο καινούριες συσκευές.

Οι απαιτήσεις του είναι:

- Pentium II ή νεότερος
- Μνήμη 64MB ή περισσότερα
- Video Display: Minimum 800x600 ανάλυση, 16 bit Color
- Σκληρός δίσκος: 10MB
- Windows 2000 Service Pack 4, Windows XP, Windows Vista 32bit/64bit
- Supported Microsoft Outlook: MS Outlook2002, MS Outlook2003, MS Outlook2007

Οι παραπάνω απαιτήσεις, όχι μόνο εξαναγκάζουν τον χρήστη να έχουν ένα συγκεκριμένο λειτουργικό, αλλά και συγκεκριμένη χρήση προγράμματος [10].

1.5 Συμπεράσματα

Όπως είδαμε, καμία από τις παραπάνω εταιρείες δεν υποστηρίζει άλλο λειτουργικό σύστημα εκτός από Windows, εξαιρουμένης της Sony Ericsson, η όποια έχει μόνο κάποια plugins για Mac λειτουργικό. Σαν αποτέλεσμα, ένας χρήστης συστήματος Linux να μην μπορεί να συνδέσει το κινητό του με κάποιο

από τα παραπάνω προγράμματα, οδηγώντας τον αναγκαστικά είτε να μην κάνει την επιθυμητή δουλειά, ή να καταφύγει σε έναν υπολογιστή με Windows. Θα πρέπει επίσης να υπενθυμίσουμε πως οι απαιτήσεις για τις εκδόσεις Windows που υποστηρίζονται θέλουν ένα σχετικά καλό μηχάνημα, ενώ εκείνα τα οποία έχουν εγκατεστημένα Vista, έχουν απαιτήσεις για έναν υπολογιστή των τελευταίων 18 περίπου μηνών. Πράγμα λίγο δύσκολο αν αναλογιστεί κανείς πως ο μέσος χρήστης έχει ένα μηχάνημα για αρκετούς μήνες παραπάνω από εκείνους που μόλις αναφέρθηκαν.

Με γνώμονα όλα όσα αναφέρθηκαν στις τελευταίες σελίδες, ο αναγνώστης μπορεί να ξεκινήσει το ταξίδι του στην Java Micro Edition και στις μεγάλες δυνατότητες αυτής της "μικρής" γλώσσας, αλλά και να κατανοήσει τους λόγους για την δημιουργία αυτής της εργασίας.

1.6 Ανακεφαλαίωση

Σε αυτό το κεφάλαιο είδαμε τους λόγους για τους οποίους αναπτύχθηκε η εφαρμογή της αυτής της πτυχιακής, καθώς ακόμα και τα προγράμματα τα οποία έχουν οι μεγάλες εταιρείες κινητών συσκευών. Στο επόμενο κεφάλαιο γίνεται μία πρώτη παρουσίαση της Java Micro Edition και της εξέλιξής της.

Κεφάλαιο 2ο

2.1 Εισαγωγή

Σε αυτό το κεφάλαιο γίνεται μία μικρή παρουσίαση της Java Micro Edition (J2ME), καθώς και μία μικρή ιστορική αναδρομή, από την αρχή της δημιουργίας της, μέχρι το σήμερα, από τα απλά JSRs μέχρι τις "ομπρέλες" προτύπων.

2.2 Java Micro Edition

Η Java Platform, Micro Edition, ή Java ME, είναι μια πλατφόρμα Java σχεδιασμένη για κινητές συσκευές και για ενσωματωμένα συστήματα. Στοχεύει σε συσκευές από βιομηχανικές συσκευές ελέγχου έως κινητά τηλέφωνα και set-top boxes. Η Java ME ήταν παλαιότερα γνωστή ως Java 2 Platform, Micro Edition (J2ME).

Java ME αντικατέστησε μία παρόμοια τεχνολογία, την Personal Java η οποία αρχικά αναπτύχθηκε υπό την Java Community Process ως Java JSR 68. Η Sun προσφέρει μια αναφορά της υλοποιήσεις της προδιαγραφής αυτής, αλλά δεν παρέχει ελεύθερα τις binary υλοποιήσεις τουJava ME Runtime Environment για κινητές συσκευές, παρά βασίζεται σε τρίτους οι οποίοι παρέχουν τις δικές τους.

Από τις 22 Δεκεμβρίου 2006, ο πηγαίος κώδικας της Java ME είναι υπό την άδεια της GNU General Public License, με την κωδική ονομασία phoneME [47].

Η Java ME τεχνολογία βασίζεται σε τρία στοιχεία:

- ένα configuration το οποίο παρέχει το βασικότερο σύνολο βιβλιοθηκών και virtual machine ικανοτήτων για ένα μεγάλο εύρος συσκευών
- ένα profile το οποίο είναι ένα σύνολο από APIs για την υποστήριξη μικρότερου εύρους συσκευών
- ένα optional package, το οποίο είναι ένα σύνολο από συγκεκριμένα APIs

[34]

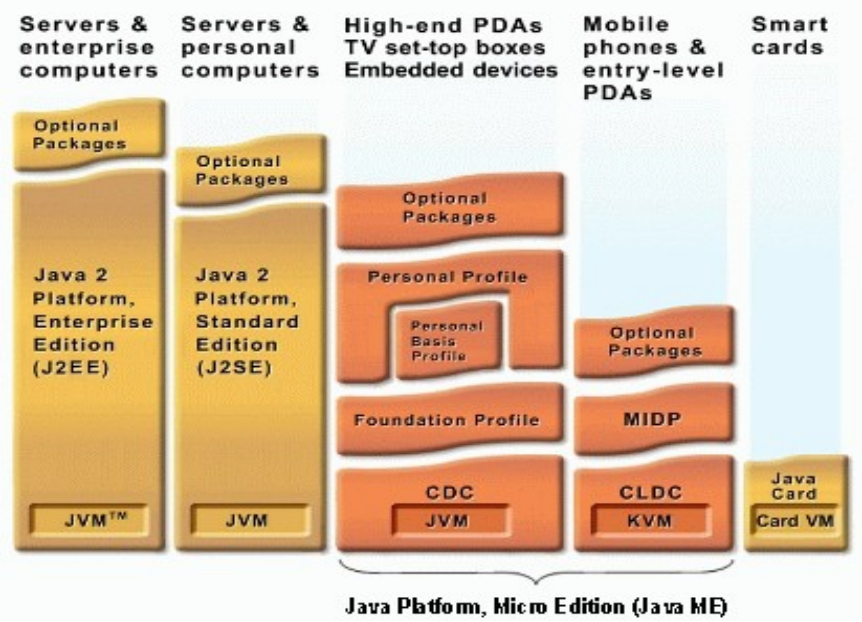
Το configuration είναι σχεδιασμένο για ένα συγκεκριμένο είδος συσκευής, με βάση τους περιορισμούς μνήμης και της ισχύς της συσκευής. Ορίζει μία JVM η οποία μπορεί εύκολα να μεταφερθεί στις συσκευές οι οποίες υποστηρίζουν το ίδιο configuration.

Το profile είναι πιο εξειδικευμένο από τα configurations. Ένα profile βασίζεται σε ένα configuration και παρέχει επιπρόσθετα APIs. όπως το user interface, η αποθήκευση δεδομένων, και οτιδήποτε άλλο είναι απαραίτητο για την ανάπτυξη εφαρμογών για τη λειτουργία της συσκευής.

Τα optional APIs καθορίζουν ειδική πρόσθετη λειτουργικότητα η οποία μπορεί να συμπεριληφθεί σε

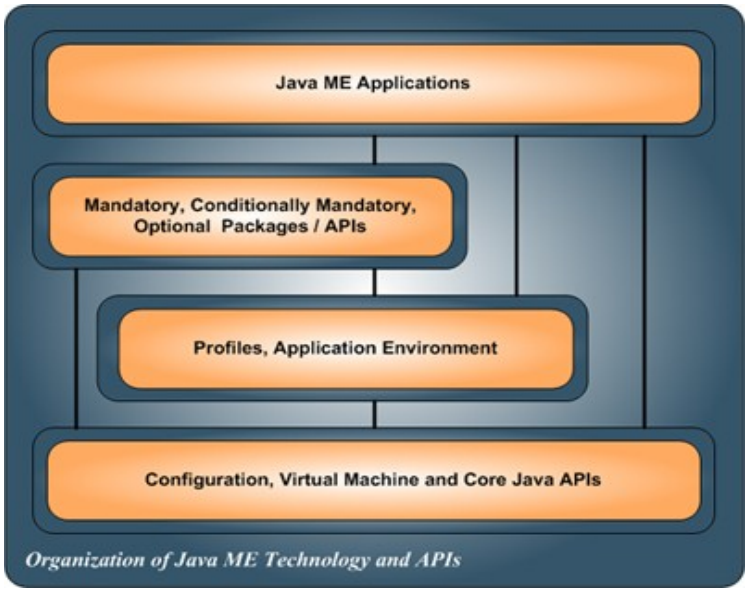
σε ένα configuration ή ένα profile.

Τα τρία παραπάνω στοιχεία, ονομάζονται stack.



Εικόνα 1.1 - J2ME Stack

[24]



Εικόνα 1.2 - Οργάνωση της Java ME τεχνολογίας και των APIs

[21]

2.3 Ιστορική αναδρομή της J2ME

Η εξέλιξη της Java Platform, Micro Edition για ξεκίνησε στις αρχές του 1998 στα εργαστήρια της Sun, με τη μορφή ενός μικρού ερευνητικού έργου. Μετά από την επίσημη καθιέρωση της Java ME platform το 1999, το ενδιαφέρον για την πλατφόρμα αυξήθηκε πολύ γρήγορα. Σήμερα, δεκάδες κατασκευαστές συσκευών σε όλο τον κόσμο είναι κατασκευάζουν συσκευές που κάνουν χρήση της Java με εκατομύρια χρήστες να τις έχουν στην κατοχή τους.

Τρία είναι τα κύρια στάδια ανάπτυξης της J2ME:

Στάδιο 1. 2000-2003: CLDC 1.0 & MIDP 1.0 συσκευές

- Οι πρώτες εκδόσεις του Connected Limited Device Configuration 1.0 (CLDC 1.0) και του Mobile Information Device Profile 1.0 (MIDP 1.0)
- Πολύ περιορισμένη μνήμη και επεξεργαστική ισχύς διαθέσιμες

Κατά την περίοδο εκείνη, έγινει πολύ γρήγορη υιοθέτηση της J2ME. Μέσα σε λίγα χρόνια 100 εκατομμύρια συσκευές υπήρχαν ήδη στην αγορά

Στάδιο 2. 2003-2006: MIDP 2.0 & και Java Technology for the Wireless Industry συσκευές

- Νέα έκδοση του προτύπου Mobile Information Device Profile (MIDP 2.0)
- Νέα έκδοση του προτύπου Connected Limited Device Configuration (CLDC 1.1) με υποστήριξη αριθμών κινητής υποδιαστολής
- Η προδιαγραφή για την Java Technoloy for the Wireless Industry (JTWI, JSR 185) μαζί με μερικά APIs κλειδιά για να δημιουργήσουν το πρώτο πρότυπο ομπρέλας με πλούσια πολυμέσα και ικανότητες επικοινωνίας
- Βελτιωμένες διαλειτουργικότητα και ικανότητες στις συσκευές
- Πολύ γρήγορη αύξηση του αριθμού των J2ME προδιαγραφών για την JTWI

Κατά την περίοδο εκείνη ο αριθμός των συσκευών από 100-200 εκατομμύρια ανέβηκε στο 1 δισεκατομμύριο συσκευών.

Στάδιο 3. 2006 μέχρι σήμερα: MSA συσκευές

- Νέα Mobile Service Architecture (MSA, JSR 248). Πρότυπο ομπρέλα θέτοντας υπό την επίβλεψή της, 16 J2ME προδιαγραφές για να θέσουν μία κανούρια βάση για την Java ME platform
- Μία νέα διαλειτουργική βάση με την δημιουργία καινούριων APIs

- Αυξημένη υιοθέτηση του πιο ικανού Connected Device Configuration (CDC) στην ασύρματη βιομηχανία.
- Νέα Mobile Service Architecture (MSA 2, JSR 249)

[1]

2.4 Ανακεφαλαίωση

Σε αυτό το κεφάλαιο έγινε μία μικρή παρουσίαση της Java Micro Edition, καθώς μία ιστορική αναδρομή από την δημιουργία της μέχρι σήμερα. Στο επόμενο κεφάλαιο ο αναγνώστης θα δει για τις διάφορες "ομπρέλες" προτύπων οι οποίες έχουν ως στόχο την ενιαία εξέλιξη κι αποδοχή των κινητών συσκευών από τους κατασκευαστές.

Κεφάλαιο 3ο

3.1 Εισαγωγή

Σε αυτό το κεφάλαιο θα δούμε την Mobile Service Architecture (MSA), την Mobile Service Architecture 2 (MSA2), το Java Technology for the Wireless Industry (JTWI) και το Generic Connection Framework (GCF). Τα πρώτα 3 αποτελούν τις λεγόμενες "ομπρέλες" πρότυπα, οι οποίες έχουν υπό την "προστασία τους" κάποια JSRs. Σκοπός τους είναι να επιτρέψουν την ενιαία εξέλιξη των κινητών συσκευών κι αποδοχή των κινητών συσκευών από τους κατασκευαστές, ώστε να έχει την δυνατότητα ο χρήστης να μπορεί να επιτελεί τις ίδιες λειτουργίες, ανεξαρτήτου μοντέλου και κατασκευαστή. Όταν μία κινητή συσκευή θεωρείται πως είναι συμβατό με την κάθε "ομπρέλα", τότε υλοποιεί όλα τα JSRs τα οποία αυτή περιλαμβάνει. Το Generic Connection Framework είναι ένα ολοκληρωμένο Framework για συνδέσεις κάθε τύπου.

3.2 Java Technology for the Wireless Industry

Η Java Technoloy for the Wireless Industry (JTWI) προδιαγραφή, JSR 185, ορίζει την πρότυπη πλατφόρμα για την επόμενη γενιά των κινητών τηλεφώνων που χρησιμοποιούν την Java τεχνολογία. Η JTWI ορίζεται μέσα από την Java Community Process (JCP)) από μία εξειδικευμένη ομάδα κατασκευαστριών εταιρειών κινητών, ανάπτυξης λογισμικού και φορείς ασυρμάτου δικτύου οι οποίοι ηγούνται στον χώρο. Η JTWI προσδιορίζει τις τεχνολογίες οι οποίες πρέπει να συμπεριληφθούν σε JTWI συμβατές συσκευές όπου αυτό είναι εφαρμόσιμο. Η προδιαγραφή ανεβάζει τον πύχη ψηλά στην λειτουργικότητα για υψηλού επιπέδου συσκευές, ενώ μειώνει τον κατακερματισμό του API και διευρύνει την ουσιαστική βάση των εφαρμογών οι οποίες έχουν ήδη αναπτυχθεί για κινητές συσκευές.

Πλεονεκτήματα

- **Διαλειτουργικότητα**: Ο στόχος αυτής της προσάθειας είναι να παραδόσει ένα προβλέψιμο περιβάλλον για προγραμματιστές εφαρμογών και ένα παραδοτέο σύνολο ικανοτήτων για κατασκευαστές συσκευών. Και οι δύο επωφελούνται σε μεγάλο βαθμό υιοθετώντας το πρότυπο JTWI: Οι κατασκευαστές από ένα μεγάλο εύρος συμβατών εφαρμογών, οι προγραμματιστές λογισμικό από ένα μεγάλο εύρος συσκευών που θα υποστηρίζουν τις εφαρμογές τους

- **Αποσαφήνιση των προδιαγραφών ασφαλείας**: Η προδιαγραφή JSR 185 εισάγει έναν αριθμό αποσαφηνίσεων για μη έμπιστες εφαρμογές όσον αφορά το "Recommended Security Policy for GSM/UMTS-Compliant Devices" το οποίο ορίζεται στην προδιαγραφή του MIDP 2.0. Επεκτείνει την βάση του MIDlet suite security framework το οποίο ορίζεται στο MIDP 2.0

- **Road map**: Ένα πολύ σημαντικό χαρακτηριστικό της JTWI προδιαγραφής είναι το road map, ένα σχεδιάγραμμα κοινής λειτουργικότητας την οποία οι προγραμματιστές εφαρμογών μπορούν να περιμένουν στις JTWI συμβατές συσκευές. Το road map δίνει την δυνατότητα σε όλους να κάνουν σχέδια για το μέλλον με περισσότερη αυτοπεποίθηση και σιγουριά: Οι φορείς μπορούν να σχεδιάσουν καλύτερα την στρατηγική ανάπτυξης εφαρμογών, οι κατασκευαστές

κινητών συσκευών θα μπορούν να κάνουν καλύτερα σχέδια και οι προγραμματιστές θα μπορούν να δουν ένα πιο ξεκάθαρο μονοπάτι για ανάπτυξη εφαρμογών.
[30]

Η JTWI αποτελείται από τα παρακάτω υποχρεωτικά και προαιρετικά JSRs:

- JSR 30: CLDC 1.0 (υποχρεωτικό)
- JSR 118: MIDP 2.0 (υποχρεωτικό)
- JSR 120: Wireless Messaging API (υποχρεωτικό)
- JSR 135: Mobile Media API (προαιρετικό)
- JSR 139: CLDC 1.1 (προαιρετικό)

[9]

3.3 Mobile Service Architecture

3.3.1 Εισαγωγή στην MSA

Η Mobile Service Architecture (MSA) πλατφόρμα χτίζει πάνω στην Java Platform, Micro Edition (Java ME) προδιαγραφές οι οποίες δημιουργήθηκαν πριν από αυτές, συμπεριλαμβανομένου και των Mobile Information Device Profile (MIDP), Connected Limited Device Configuration (CLDC), and Java Technology for the Wireless Industry (JTWI). Καθώς η αγορά των ασυρμάτων συσκευών συνεχίζει να εξελίσεται και να ενσωματώνει νέες τεχνολογίες κι υπηρεσίες, υπάρχει η απαίτηση για την δημιουργία μίας πλατφόρμας η οποία να προτυποποιεί πάνω σε αυτές τις νέες τεχνολογίες. Η ευρεία υιοθέτηση κι επιτυχία του MIDP και CLDC αγορά των κινητών συσκευών, η οποία ακολουθήθηκε από την επιτυχία της JTWI πλατφόρμας, πλέον αντικαθίσταται από το επόμενο πρότυπο στην βιοχημανία ασύρματων συσκευών, την MSA [28].

Τα JSR 248 και JSR 249 ορίζουν την καινούρια γενιά πάνω στην πλατφόρμα της Java γα τις κινητές συσκευές. Αυτά τα δύο ορίζουν μία περιεκτικότατη δομή από APIs με στόχο την διευκόλυνση ανάπτυξης της μεγαλύτερης δυνατής ποικιλίας εφαρμογών, σε μία μορφή που θα είναι εύκολα φορητή στο ευρύτερο δυνατό φάσμα κινητών συσκευών.

Το JSR 249, Mobile Service Architecture Advanced βασισμένο σε μία ανανέωση του Connected Device Configuration (CDC), θα υποστηρίζει εφαρμογές οι οποίες χρειάζονται τους πόρους των πιο υπολογιστικά ισχυρών συσκευών στην αγορά, όπως τα PDA.

Το JSR 248 βασισμένο στο Connected Limited Device Configuration (CLDC) θα είναι απαραίτητα πιο ευέλικτο και αποτελεσματικό, αλλά προσδιορίζει ένα μεγάλο υποσύνολο των σταθερών ικανοτήτων του JSR 249.

Όπως προϊδεάζει και το όνομά της, η MSA είναι μία ολοκληρωμένη αρχιτεκτονική για κινητές συσκευές. Όπως ο προκάτοχός του, JSR 185: Java Technology for the Wireless Industry, η MSA είναι

μία ομπρέλα για μία συλλογή της ίδιας οικογενείας ανανεωμένων και καινούριων JSR τα οποία συνεργάζονται για να υποστηρίζουν εφαρμογές με ένα μεγάλο εύρος προτυποποιημένων ικανοτήτων. Διευρύνει την αρχιτεκτονικη που οριζόταν από το JSR 185, για να ενσωματώσει νέες τεχνολογίες για υψηλού επιπέδου κινητές συσκευές.

Κάποια από τα JSR είναι υποχρεωτικό, ενώ άλλα είναι υπό όρους υποχρεωτικά. Για να συμβαδίζει με την MSA μία εφαρμογή, πρέπει να υποστηρίζει ένα JSR αν είναι υποχρεωτικό ή αν είναι υπό όρους υποχρεωτικό και οι όροι είναι αληθείς [20].

3.3.2 MSA API

Η MSA σκοπεύει να μειώσει τον κατακερματισμό της Java community με δύο τρόπους: Η MSA ορίζει ένα ξεκάθαρο σύνολο από JSRs τα οποία θα πρέπει να υποστηρίζονται (υποχρεωτικά JSRs) στις επόμενες γενιές κινητών τηλεφώνων. Η MSA δίνει λιγότερο χώρο για διευρμήνευση και υλοποοίηση των JSRs αποσαφηνίζοντας τις συνθήκες για αλληλεπίδραση μεταξύ αυτών και των προαιρετικών JSRs. Τα υποχρεωτικά APIs που υποστηρίζονται από την MSA περιλαμβάνουν 3D γραφικά, διαχείριση προσωπτικών πληροφοριών, Bluetooth™, Scalable Vector Graphics (SVG), web υπηρεσίες, χωρικές υπηρεσίες και υπηρεσίες πληρωμών για την Java ME (Micro Edition). Η Sony Ericsson έχει υποστηρίξει μερικά από αυτά τα APIs, σε προηγούμενες εκδόσεις της Java Platform, αλλά με την MSA, υπάρχει επιτέλους μία κοινή γραμμή πορείας στην βιομηχανία και ένα ξεκάθαρο σύνολο προσδοκιών στην υλοποίηση αυτών των JSRs. Με την MSA, η μαζική αγορά κινητών τηλεφώνων γίνεται ικανή πλατφόρμα για ανάπτυξη εμπορικών εφαρμογών και θα είναι δυνατό να διασφαλίσει ασφαλή αποθήκευση δεδομένων και επικοινωνία και να δημιουργήσει καινοτόμες εφαρμογές. Η MSA μειώνει τα διαφορετικά περιβάλλοντα και δημιουργεί ένα προβλέψιμο περιβάλλον για προγραμματιστές, μειώνοντας τα θέματα μεταφερσιμότητας και δημιουργόντας νέες εφαρμογές και παιχνίδια για κινητά με πλούσια χαρακτηριστικά στην μαζική αγορά [2].

3.3.3 Λειτουργικότητα και υποχρεωτικά JSRs

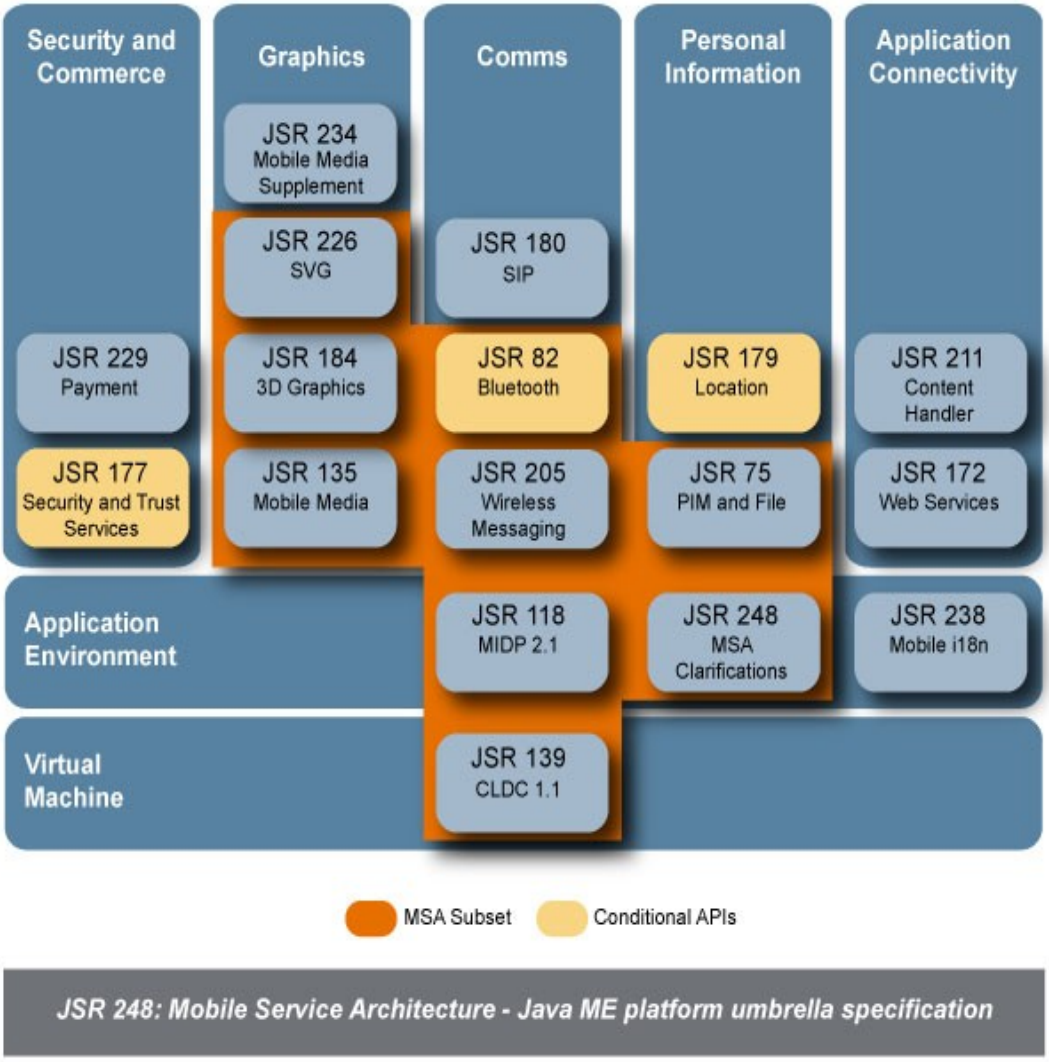
Οι στόχοι της MSA προδιαγραφής είναι να καθοριστεί ένα πρότυπο σύνολο λειτουργιών για εφαρμογές κινητών συσκευών ενώ θα αποσαφηνίσει αλληλεπιδράσεις μεταξύ των διαφόρων τεχνολογιών που συνδέονται με τις MIDP και CLDC και προδιαγραφές. Επειδή υπάρχει ένα ευρύ φάσμα διακύμασνης στις ικανότητες υλικού και λογισμικού στα κινητά η MSA την προδιαγραφή προσφέρει δύο επιλογές: να υλοποιήσεις το προκαθορισμένο υποσύνολο της MSA προδιαγραφής ή να υλοποιήσει την πλήρη MSA προδιαγραφή. Οι συμβατές με την MSA συσκευές πρέπει να υλοποιούν είτε όλο το προκαθορισμένο υποσύνολο ή την πλήρη MSA προδιαγραφή:

- Το υποσύνολο καλύπτει την βασική λειτουργικότητα
- Η πλήρης προδιαγραφή στοχεύει στις πλούσιες σε χαρακτηριστικά τελευαίας τεχνολογίας κινητά

Όσο περνάνε τα χρόνια, η MSA θα γινεται για όλο και περισσότερες συσκευές το βασικό περιβάλλον.

Το υποσύνολο της MSA περιλαμβάνει μερικά JSRs: τα 139, 118, 75, 135, 82, 184, 205 και 226. Η

προδιαγραφή της πλήρους MSA περιλαμβάνει όλα τα JSRs του υποσυνόλου της, καθώς και τα: 172, 180, 211, 229, 234, 238, 177 και 179. Ενώ τα περισσότερα JSRs είναι υποχρεωτικά, λίγα JSRs είναι υπό όρους υποχρεωτικά, επειδή εξαρτάται από το αν η συσκευή παρέχει την ικανότητα του υλικού για να μπορέσει να υποστηρίξει αυτά τα JSRs. Επιπλέον, η MSA προδιαγραφή ορίζει συμπληρωματικές διευκρινίσεις για τις προδιαγραφές των JSRs συχνά προσθέτωντας μέχρι πρότεινως προαιρετικά κομμάτια μέσα σε αυτές. Ο σκοπός είναι να δημιουργηθεί ένα πιο προβλέψιμο περιβάλλον για τις κινητές συσκευές.



Εικόνα 3.1 - MSA – JSR 248

3.3.4 MSA Subset

Για την υλοποιηθεί το υποσύνολο της MSA, οι συσκευές πρέπει να υποστηρίζουν το CLDC (JSR 139) και το MIDP (JSR 118), όπως και τα JSR 75, 135, 184, 205 και 226. Εάν η συσκευή υποστηρίζει Bluetooth, θα πρέπει επίσης να υποστηρίζει το JSR 82. Επιπλέον, οι συσκευές πρέπει να υποστηρίζουν ορισμένες διευκρινίσεις για τα JSRs όπως αυτά περιγράφονται στην MSA προδιαγραφή. Με αυτό το σύνολο των προδιαγραφών, οι προγραμματιστές μπορούν να δημιουργήσουν μια μεγάλη ποικιλία από εφαρμογές οι οποίες είναι ικανές να κάνουν χρήση πολυμέσων αλλά και των Personal Information Manager (PIM) δεδομένων. Όταν η MSA χρησιμοποιείται για πρώτη φορά, οι επιχειρηματίες και οι κατασκευαστές συσκευών μπορούν να επιλέξουν να εφαρμόσουν το υποσύνολο της MSA ως κοινή βάση σε όλες τις συσκευές. Καθώς οι ικανότητες του υλικού αυξάνονται σε όλες τις συσκευές, η προδιαγραφή της πλήρης MSA θα πρέπει να καταστεί πιο διαδεδομένη από το υποσύνολο της MSA.

JSR 139: CLDC 1.1 (υποχρεωτικό)

Το βασικό configuration για την MSA είναι το CLDC 1.1. Αυτή η προδιαγραφή βασίζεται σε μια virtual machine σχεδιασμένη, έχοντας κατά νου της συσκευές με περιορισμένους πόρους. Παρέχει βασική τεχνολογία για την Java ME runtime platform και τις βιβλιοθήκες, και χρησιμοποιείται ως η βάση για ένα ή περισσότερα profiles, συμπεριλαμβανομένου και του MIDP.

JSR 181: MIDP 2.1 (υποχρεωτικό)

Το MIDP παρέχει την βασική λειτουργικότητα για εφαρμογές κινητών συσκευών, συμπεριλαμβανομένου του user interface, τη δυνατότητα σύνδεσης στο δίκτυο, της τοπική αποθήκευσης δεδομένων, και την διαχείριση του life cycle της εφαρμογής. Απευθύνεται σε κινητές συσκευές όπως ασύρματα τηλέφωνα ή PDAs.

JSR 75: PDA Optional Packages για την J2ME™ platform

Το JSR 75 ορίζει δύο πακέτα για την Java ME platform, δύο από τα οποία είναι υποχρεωτικά. Το PIM πακέτο επιτρέπει στους προγραμματιστές να έχουν πρόσβαση στα PIM δεδομένα, όπως το ημερολόγιο, βιβλίο διευθύνσεων, και οι λίστες υποχρεώσεων. Το File Connection πακέτο στους προγραμματιστές να έχουν πρόσβαση σε διάφορες μορφές των δεδομένων, όπως εικόνες, ήχους, βίντεο, και άλλα, στο file system των κινητών συσκευών. Αυτό περιλαμβάνει αφαιρούμενες μονάδες αποθήκευσης, όπως οι κάρτες μνήμης. Η πρόσβαση και στα δεδομένα PIM και στα αρχεία επιτρέπει στις εφαρμογές να μπορεί να ελέγχει τις πληροφορίες της εκάστοτε συσκευής, δίνοντάς τους, την δυνατότητα να είναι πιο έξυπνες κι εύκολες προς χρήση.

JSR 135: Mobile Media API 1.2 (υποχρεωτικό)

Το MMAPI επιτρέπει την πρόσβαση και τον έλεγχο, συμπεριλαμβανομένων αναπαραγωγή και τη σύλληψη, βασικού ήχου και πολυμέσων σε κινητές συσκευές. Αυτό παρέχει ένα πλούσια σε media, προς τον χρήστη, εμπειρία, ενώ με αυτό τον τρόπο επεκτείνεται η πλατφόρμα στα πολυμέσα. Ως πολύ μικρό, ελαφρύ πακέτο, το MMAPI βοηθά προγραμματιστές έχουν πρόσβαση στις υπηρεσίες

πολυμέσων των συσκευών.

JSR 82: APIs για Bluetooth 1.1 (υπό όρους υποχρεωτικό)

Το JSR 82 επιτρέπει στους προγραμματιστές να δημιουργήσουν εφαρμογές οι οποίες χρησιμοποιούν την τεχνολογία Bluetooth, ένα ευρέως ανεπτυγμένο πρότυπο για ασύρματη επικοινωνία. Το bluetooth μπορεί να χρησιμοποιηθεί για την ανταλλαγή αρχείων, φωτογραφιών, business cards ή άλλα στοιχεία μεταξύ κινητών συσκευών. Αν η συσκευή υποστηρίζει Bluetooth, τότε πρέπει επίσης να υποστηρίζει JSR 82, προκειμένου να είναι συμβατή με την MSA.

JSR 184: Mobile 3D Graphics για J2ME 1.1 (υποχρεωτικό)

Η Mobile 3D Graphics API προδιαγραφή επιτρέπει τη χρήση εξελιγμένων 3D γραφικών σε ένα ελαφρύ, διαδραστικό περιβάλλον με μικρές ROM και RAM. Τα 3D γραφικά χρησιμοποιούνται σε μια ευρεία ποικιλία εφαρμογών για κινητές συσκευές, από παιχνίδια μέχρι screen savers και μηνυμάτα. Το JSR 184 μπορεί να υλοποιηθεί τόσο σε υψηλού αλλά και χαμηλού επιπέδου πλατφόρμες όλα τα κινητά τηλέφωνα στην αγορά.

JSR 205: Wireless Messaging API 2.0 (υποχρεωτικό)

Το Wireless Messaging API (WMA) 2.0 είναι ένα υπερσύνολο του WMA 1.0 (JSR 120). Δίνει στους Java ME προγραμματιστές τη δυνατότητα να στέλνουν και να λαμβάνουν SMS, MMS και CBS μηνύματα. Επειδή η αποστολή μηνυμάτων είναι μία από τις πιο δημοφιλείς χρήσεις για των κινητών τηλεφώνων, το WMA επιτρέπει στους προγραμματιστές να ενσωματώσει τα μηνύματα μέσα σε εφαρμογές σε αυτές τις συσκευές.

JSR 226: Scalable 2D Vector Graphics API για J2ME 1.1 (υποχρεωτικό)

Η Scalable 2D Vector Graphics API προδιαγραφή καθορίζει ένα API για το rendering 2D γραφικών στο World Wide Web Consortium (W3C) στην Scalable Vector Graphics (SVG) Tiny μορφή. Το SVG καθιστά δυνατόν για τους προγραμματιστές τη δημιουργία διαδραστικών γραφικών, με τη δυνατότητα να κάνουν ζουμ και να αλλάξουν το μέγεθος της οθόνης και τις αναλύσεις της. Το JSR 226 ορίζει επίσης ένα υποσύνολο του Micro Document Object Model (uDOM) API για να επιτρέψει την αλληλεπίδραση του χρήστη και την δυναμική χειραγώγηση του SVG περιεχομένου. Οι εφαρμογές που εκμεταλλεύονται το JSR 226 μπορούν να αποθηκεύουν, να φορτώνουν, να διαχειρίζονται και να κάνουν rendering, 2D γραφικών, δημιουργώντας μια πιο δυναμική και πλούσια εμπειρία για τον χρήστη. Με το JSR 226, οι προγραμματιστές μπορούν να επωφεληθούν του μεγάλου ποσού του διαθέσιμου SVG περιεχομένου. Επειδή το SVG στηρίζεται στην XML, οι προγραμματιστές που έχουν συνηθίσει σε ένα scripting περιβάλλον μπορούν επίσης να επωφεληθούν από τις ισχυρές δυνατότητες που προσφέρονται από έναν πλήρως προγραμματιστικό περιβάλλον, όπως αυτό της Java.

3.3.5 Πλήρης MSA προδιαγραφή

Προκειμένου να συμμορφωθούν με την πλήρη MSA προδιαγραφή, οι υλοποιήσεις της θα πρέπει να συμμορφώνονται με όλες τις απαιτήσεις υποσυνόλου της MSA καθώς με τις απαιτήσεις και δικευκρινίσεις από τα ακόλουθα JSRs. Η προδιαγραφή της πλήρους MSA απευθύνεται σε high-end κινητές συσκευές. Επίσης, παρέχει ένα roadmap ώστε μία κοινή πλατφόρμα να χρησιμοποιηθεί από την μαζική αγορά κινητών συσκευών στο όχι πολύ μακρινό μέλλον. Με τη MSA προδιαγραφή, οι φορείς και οι κατασκευαστές μπορούν να αναπτύξουν μια στιβαρή και προβλέψιμη βασική πλατφόρμα η οποία να επιτρέπει στους προγραμματιστές να ενσωματώσουν προηγμένες δυνατότητες υλικού και λογισμικού σε εξελιγμένες διαδραστικές εφαρμογές.

JSR 172: J2ME Web Services προδιαγραφή 1.0 (υποχρεωτικό)

Η J2ME Web Services προδιαγραφή επιτρέπει στους προγραμματιστές να επωφεληθούν από τις υφιστάμενες Web υπηρσίες συμβάσεις όταν δημιουργούν πελάτες για τις υπηρεσίες επιχειρήσεων. Περιλαμβάνει έναν XML parser JAXP καθώς και τα API τα οποία δίνουν τη δυνατότητα για Remote Procedure Call (RPC)επικοινωνία. Η J2ME Web Services προδιαγραφ ακολουθεί τις συμβάσεις και τα APIs των προδιαγραφών των Web υπηρεσιών οι οποίες χρησιμοποιούνται από την J2SE και J2EE. Με το JSR 172, οι προγραμματιστές μπορούν να χρησσιμοποιήσουν ήδη υπάρχουσες έννοιες του Web έννοιες με σκοπό την γρήγρορη ανάπτυξη και χρήση Web υπηρεσιών.

JSR 180: SIP API για J2ME 1.0.1 (υποχρεωτικό)

Το Session Initiation Protocol (SIP) API για J2ME επιτρέπει στους προγραμματιστές να ξεκινήσουν να απαντήσουν, και ο χειριστούν SIP αιτήσεις σε συκευές με περιορισμένη μνήμη, όπως τα κινητά τηλέφωνα. Το SIP API βασίζεται στο υποκείμενο CLDC Generic Connection Framework και ακολουθεί το μοντέλο ασφαλείας της MSA. Οι SIP εφαρμογές επιτρέπουν λειτουργικότητες όπως άμεσα μηνυμάτα, παιχνίδια, chatting. Καθώς το υποκείμενο κινητό IP περιβάλλον γίνεται όλο και πιο εξελιγμένο, το SIP API θα επιτρέπει στα MIDlets να επωφελούνται από την αυξημένη συνδεσιμότητα επιτρέποντας σσταθερές σε πραγματικό χρόνο ενημερώσεις δεδομένων.

JSR 211: Content Handler API 1.0 (υποχρεωτικό)

Το Content Handler API (CHAPI) επιτρέπει στις Java ME εφαρμογές να καλούν μία άλλη εφαρμογή να χειριστούν συγκεκριμένου τύπου περιεχόμενο. Η κατάλληλη εφαρμογή για να χειριστεί τον τύπο ενός περιεχομένου βασίζεται στο appilcation management system (AMS) της εφαρμογής τηςσυσκευής. Οι εφαρμογές μπορούν να καταχωρίσουν τον εαυτό τους με το AMS ώστε να χειριστούν διαφόρους τύπους περιεχομένου. Με το CHAPI, μπορούν να αναπτυχθούν εφαρμογές οι οποίες ανάλογα με τον τύπο των δεδομένων θα επιτελούν κάποια ενέργεια. Πχ αν ένα μήνυμα έχει μουσική, τότε το CHAPI θα καλέσει τον music player της συσκευής.

JSR 229: Payment API 1.1.0 (υποχρεωτικό)

Το Payment API επιτρέπει στις Java ME εφαρμογές να κάνουν αίτηση, αλλά και να ξεκινήσουν συναλλαγές, όπως η αγορά ring tones, ή wallpapers για τις κινητές συσκευές τους. Ορίζει επίσης μία σύνταξη για την περιγραφή των δεδομένων τα οποία υποστηρίζουν διάφορα συστημάτα πληρωμών. Αυτό το ευέλικτο API υποστηρίζει πολλαπλές μεθόδους πληρωμής, όπως πληρωμή συνδρομητικών υπηρεσιών που χρεώνονται από τους φορείς εκμετάλλευσης, μονής φοράς πληρωμές

προς τρίτους.

JSR 234: Advanced Multimedia Supplements (υποχρεωτικό)

Το Advanced Multimedia Supplements έχει ως στόχο για την επέκταση του MMAPI παρέχοντας πρόσβαση σε χαρακτηριστικά που έχουν γίνει αρκετά κοινά στις κινητές συσκευές, όπως οι κάμερες, τα 3D αρχεία ήχου, το ραδιόφωνο, και η κωδικοποίηση εικόνας. Το JSR 234 επιτρέπει στους προγραμματιστές να έχουν πρόσβαση στις πολυμεσικές ικανότητες τς συσκευής, καθιστώντας ικανές τις εφαρμογές για τον έλεγχό τους. Άλλες εφαρμογές μπορεί να ενσωματώσουν πολυμέσα, όπως η αποστολή μηνυμάτων επιτρέποντας στους χρήστες να στέλνουν εικόνες από τις κάμερές τους.

JSR 238: Mobile Inernationalization API 1.0 (υποχρεωτικό)

Το JSR 238 επιτρέπει την διεθνοποίηση των Java ME εφαρμογών, καθιστώντας δυνατή την απομόνωση πόρων και την χρήση τους κατά την εκτέλεση ανάλογα με τον γεωγραφικό τόπο στον οποίο βρίσκεται ο χρήστης. Αυτό επιτρέπει στον προγραμματιστή να δημιουργήσει εύκολα μία εφαρμογή σε πολλές γλώσσες, με διαφορετικές ώρες, με διαφορετικό τρόπο μέτρησης για γεωγραφικά σημεία του πλανήτη.

JSR 177: Security and Trust Services API για J2ME 1.0 (υποχρεωτικό, υπό όρους υποχρεωτικό, προαιρετικό)

Το Security and Trust Services API (SATSA) παρέχει πρόσβαση σε υπηρεσίες ασφαλείας για κινητές συσκευές, συμπεριλαμβανομένου της ασφαλούς αποθήκευσης ευαίσθητων δεδομένων, κρυπτογραφικές λειτουργίες για την υποστήριξη συναλλαγών πληρωμής και την ακεραιότητα δεδομένων, καθώς και ένα ασφαλές περιβάλλον εκτέλεσης το οποίο επιτρέπει την ανάπτυξη χαρακτηριστικών ασφαλείας. Οι προγραμματιστές μπορεί να βασίζονται σε αυτές τις υπηρεσίες για να υποστηρίξουν μία μεγάλη ποικιλία εφαρμογών. Από εφαρμογές που αφορούν τραπεζικές συναλλαγές και πληρωμές, την αυθεντικοποίηση και ταυτοποίηση των χρηστών.

Το SATSA αποτελείται από τέσσερα πακέτα:

- SATSA-crypto (υποχρεωτικό) για την υποστήριξη κρυπτογραφικών λειτουργιών όπως είναι η συνόψιση μηνύματος, επιβεβαίωση υπογραφής και η κρυπτογράφηση ή αποκρυπτογράφηση
- SATSA-APDU (υπό όρους υποχρεωτικό) για επικοινωνία με εφαρμογές smart card
- SATSA-PKI (υπό όρους υποχρεωτικό) για την υποστήριξη ψηφιακών υπογραφών
- SATSA-JCRMI (προαιρετικό) για την υποστήριξη remote invocation μιας Java Card ™ συσκευή

Πέρα από το μοντέλο ασφαλείας το οποίο ήδη υπάρχει στις Java ME εφαρμογές, το SATSA επιτρέπει στους προγραμματιστές να προσθέσουν εύκολα υπηρεσίες ασφαλείας οι οποίες υποστηρίζουν χαρακτηριστικά, όπως η αναγνώριση των χρηστών, η ακεραιότητα των δεδομένων, καθώς και συναλλαγές πληρωμής.

JSR 179: Location API για J2ME 1.0.1 (υπό όρους υποχρεωτικό)

Το JSR 179 επιτρέπει συσκευές τεχνολογίας Java ME να καθορίσουν την φυσική τους τοποθεσία, ενεργοποιώντας εφαρμογές που αναγνωρίζουν την τοποθεσία. Παραδείγματα περιλαμβάνουν την χαρτογράφηση, εφαρμογές μηνυμάτων. [29]

3.4 Mobile Service Architecture 2

3.4.1 Εισαγωγή στην MSA2

Η Java ME κοινότητα έχει αναπτύξει ένα ενιαίο, πρότυπο Java περιβάλλον για εφαρμογές για τα κινητά τηλέφωνα ως τμήμα της προδιαγραφής JSR 248 Mobile Service Architecture (MSA). Η MSA2 (JSR 249) συνεχίζει το έργο της προηγούμενης, απευθυνόμενη σε ένα ακόμα ευρύτερο σύνολο από συσκευές, σε σχέση με την MSA, με περισσότερες και ποικίλες ικανότητες στοχεύοντας στις υψηλού επιπέδου συσκευές, οι οποίες θα βρίσκονται στην αγορά στο πολύ κοντινό μέλλον. Η MSA2 προδιαγραφή ορίζει ένα σύνολο τεχνολογιών Java ME και δείχνει πώς αυτές οι τεχνολογίες πρέπει να είναι σωστά ενσωματωμένες σε μια φορητή συσκευή ώστε να δημιουργήσει μία βέλτιστη κινητή Java platform.

Στόχοι σχεδιασμού

Ο πρωταρχικός στόχος του σχεδιασμού της MSA2 προδιαγραφής είναι να ελαχιστοποιηθεί ο κατακερματισμός των Java κινητών περιβαλλόντων, ορίζοντας μία προβλέψιμη και διαλειτουρική εφαρμογή, καθώς κι ένα περιβάλλον υπηρεσίας για προγραμματιστές, τα οποία θα διασφαλίσουν την λειτουργικότητα των εφαρμογών στις συσκευές οι οποίες την υποστηρίζουν.

Ο δεύτερος στόχος για το σχεδιασμό της MSA2 είναι να επιτρέψει τη χρήση της σε μια μεγάλη ποικιλία από διαφορετικές αγορές και πελάτες. Αυτό επιτυγχάνεται επιτρέποντας στο CDC να είναι το υποκείμενο configuration και σαφώς καθορισμένες προϋποθέσεις για τις δυνατότητες που μπορεί να μην είναι διαθέσιμες σε όλα τα συμβατά με την MSA2 προϊόντα.

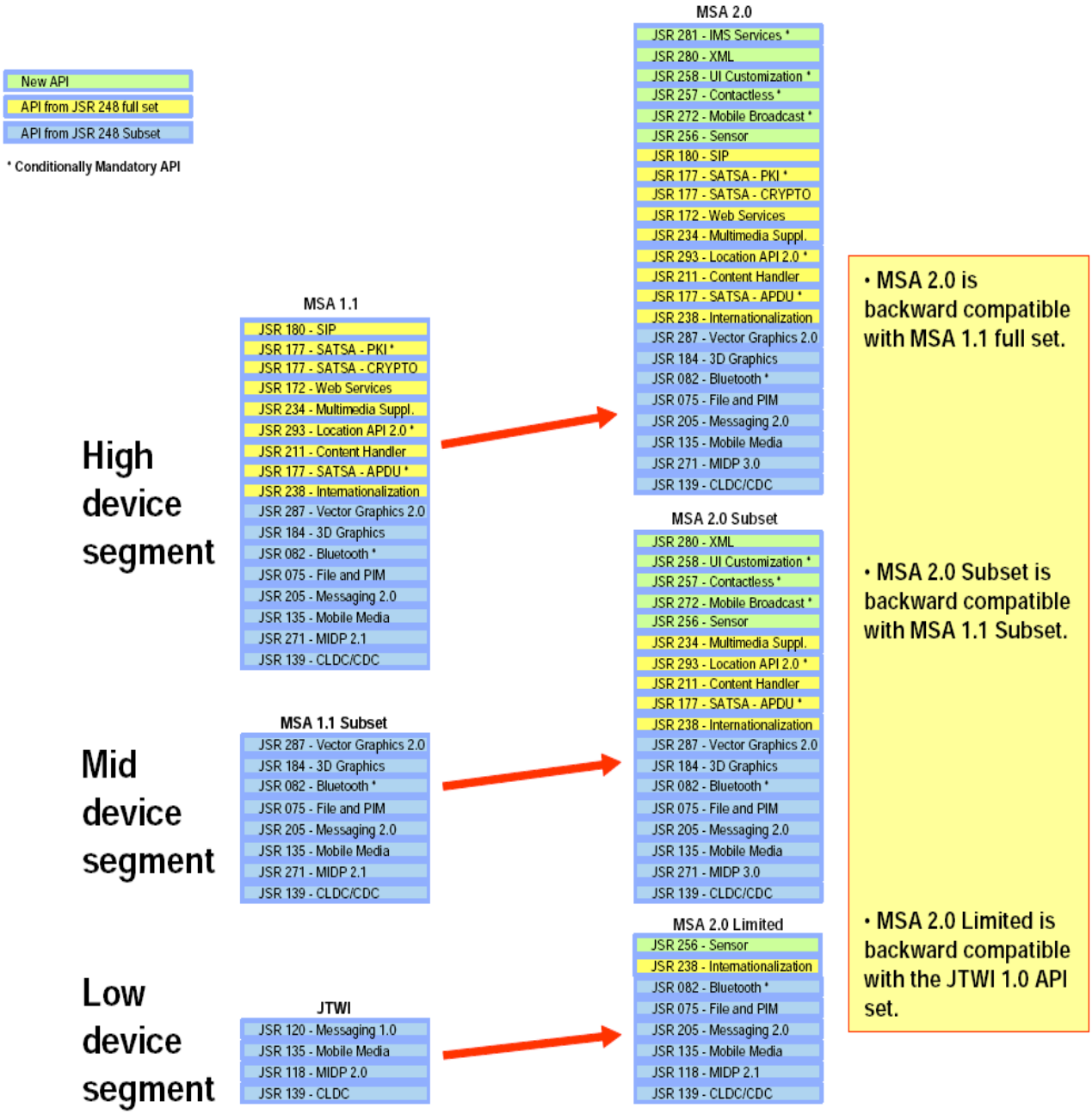
Ο τρίτος στόχος του σχεδιασμού της MSA2 είναι να παρέχει συμβατότητα προς τα πίσω προς την MSA και την JTWI.

3.4.2 Τα συστατικά μέρη της MSA2

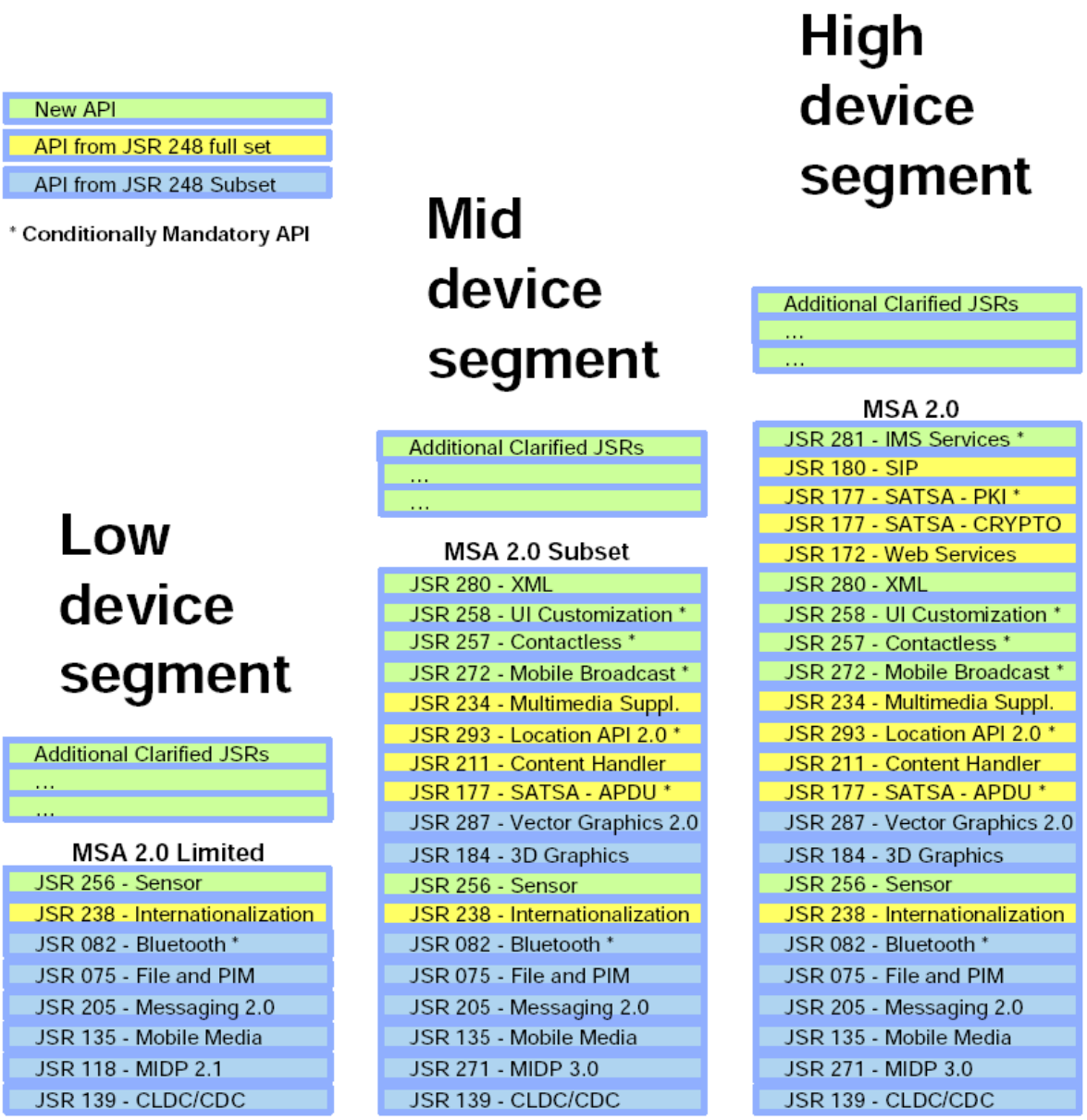
Η MSA2 έχει μία νέα εμφάνιση, αλλά παραμένει προς τα πίσω συμβατή με την προκάτοχό της. Εκεί όπου η MSA προσέφερε δύο εκδόσεις – ολόκληρη (full) και υποσυνόλου (subset) – η MSA2 προσφέρει τρεις εκδόσεις: Περιορισμένη (Limited), Υποσυνόλου (Subset) και Πλήρης (Full).

- Η MSA2 Limited έκδοση ευθυγραμμίζεται με το JSR 185, Java Technology for the Wireless Industry, το οποίο ήταν ο προκάτοχος της MSA
- Το υποσύνολο της MSA2 ευθυγραμμίζεται με εκείνο της MSA
- Η MSA2 Full ευθυγραμμίζεται με την MSA Full

Κάθε έκδοσης της MSA2 είναι υπερσύνολο της αντίστοιχης MSA.



Εικόνα 3.2 - MSA Evolution Path



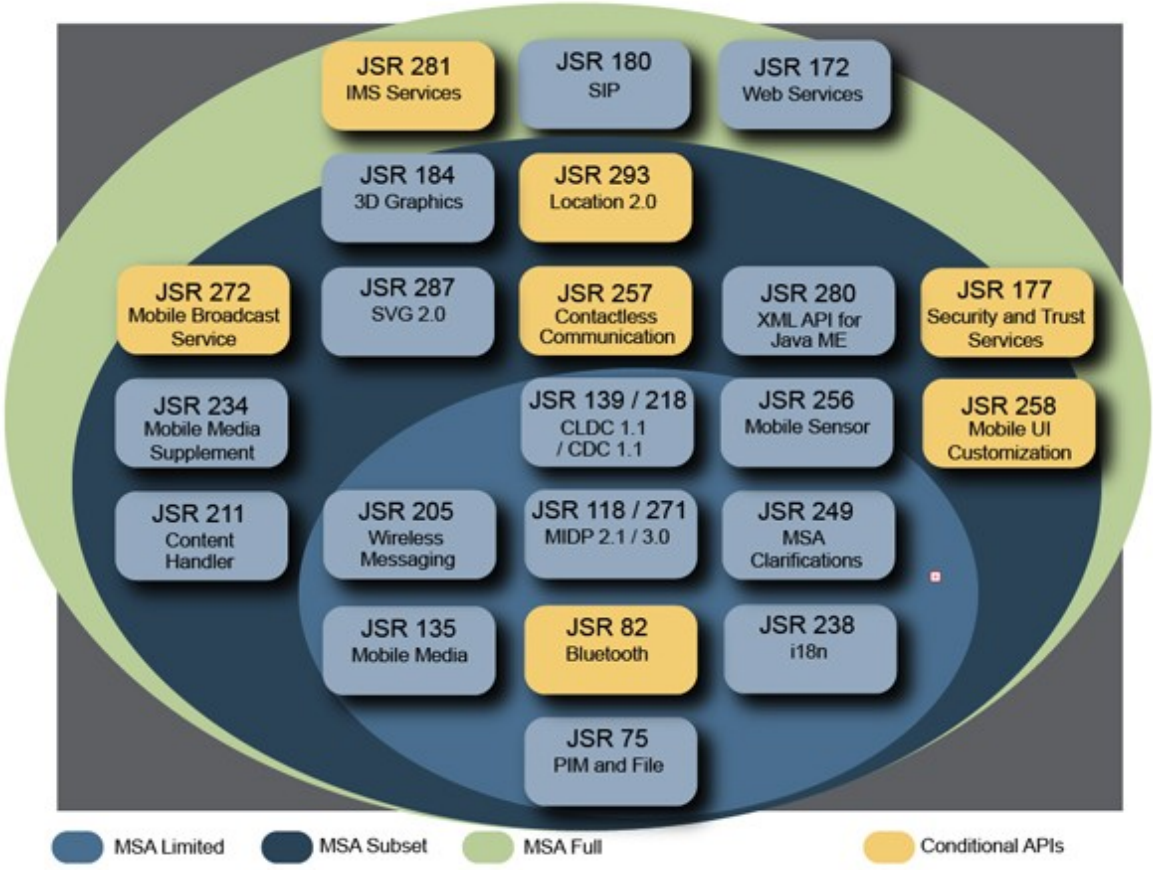
Εικόνα 3.3 - Τα συστατικά μέρη της MSA2

Μια συμβατή υλοποίηση της MSA2 πρέπει να πληρεί τις ακόλουθες προϋποθέσεις:

- Να υλοποιηθεί κάθε JSR ενός από τα σύνολα API (MSA 2.0 Limited, το υποσύνολο της MSA 2.0 ή MSA 2.0) τα οποία αναφέρονται στον παρακάτω πίνακα. Αυτό περιλαμβάνει την υλοποίηση κάθε υπό όρους υποχρεωτικού JSR εάν υπό όρους την εφαρμογή κάθε υποχρεωτική εάν το JSR αντιστοιχεί ο όρος.

- Συμμόρφωση με όλες τις πρόσθετες απαιτήσεις που εφαρμόζονται για όλα τα JSRs (συμπεριλαμβανομένων και των μη υποχρεωτικών JSRs).

- Συμμόρφωση με όλες τις επιπρόσθετες απαιτήσεις.



Εικόνα 3.4 - MSA2 – JSR 249

3.4.3 Νέα χαρακτηριστικά

Στην εικόνα επάνω, γίνονται αμέσως εμφανείς δύο αλλαγές.

- **Virtual Machine layer:** Μία συσκευή MSA2 μπορεί να χρησιμοποιήσει είτε το CLDC 1.1 είτε το CDC 1.1 σαν το configuration της. Αυτή η αλλαγή επεκτείνει το περιβάλλον εκτέλεσης του MIDlet στο Connected Device Configuration

- **Application Environment level:** Η MSA2 Limited χρησιμοποιεί το MIDP 2.1 ενώ η MSA2 Subset και η MSA2 Full χρησιμοποιούν το MIDP 3.0

JSR 218: Connected Device Configuration (CDC) 1.1 (υποχρεωτικό)

Με την MSA2 πλέον υπάρχουν δύο υποστηριζόμενα configurations: Τα CLDC 1.1 και CDC 1.1. Αυτό έγινε για να μπορούν να το χρησιμοποιούν τα MIDlets κι έτσι να αποτελέσει μία βάση για τις ήδη υπάρχουσες εφαρμογές στις επόμενες Java ME platoforms.

JSR 229: Payment API (έχει αφαιρεθεί)

Το Payment είναι το μόνο JSR το οποίο ήταν μέρος της MSA και δεν είναι της MSA2.

JSR 256: Mobile Sensor API (υποχρεωτικό)

Το Mobile Sensor επιτρέπει πρόσβαση στους αισθητήρες της συσκευής. Σαν αποτέλεσμα αυτοί μπορούν να συλλέγουν δεδομένα τα οποία μπορούν να χρησιμοποιηθούν από μια εφαρμογή για κάτι τόσο απλό όπως τον έλεγχο ενός παιχνιδιού μέχρι και την χρήση βιομετρικών στοιχείων πολύπλοκων περιβαλλοντικών ελέγχων.

JSR 257: Contactless Communication API (υποχρεωτικό)

Τα Think Radio Frequency Identification (RFID), Near Field Communication (NFC), και τα bar codes είναι μικρές ποσότητες δεδομένων οι οποίες διαβάζονται από τα αντικείμενα γύρω μας. Εάν κάποιος θέλει να κάνει σύγκριση των προϊόντων γύρω του, τότε μπορεί με την κινητή του συσκευή να διαβάσει το bar code του προϊόντος, και να συγκρίνει τις τιμές με τα αντίστοιχα προϊόντα σε on-line καταστήματα.

JSR 258: Mobile User Interface Customization API (υποχρεωτικό)

Σχεδόν όλες οι ηλεκτρονικές συσκευές υποστηρίζουν κάποιου είδους user interface όσον αφορά την τα χρώματα, την γραμματοσειρά, και το σχήμα των user interface στοιχείων. Αυτό το JSR επιτρέπει στους χρήστες να δημιουργήσουν το δικό τους look.

JSR 271: MIDP 3.0 (υποχρεωτικό)

Αυτή θα είναι η 3η μεγάλη αναθεώρηση του MIDP και σαν αποτέλεσμα έρχεται με αρκετά νέα χαρακτηριστικά. Το MIDP3 θα υποστηριχθεί από δύο CLDC και CDC virtual machines. Θα επιτρέπεται ο MIDlet συγχρονισμός ο οποίος θα προσφέρει multi-tasking ελευθερίες στα MIDlets, όπως και η διαεπικοινωνία μεταξύ τους. Το νέο πακέτο `javax.microedition.event` θα προσφέρει επικοινωνία από εφαρμογή σε εφαρμογή και σύστημα κοινοποίησης των events.

JSR 272: Mobile Broadcast Service API για Handheld Terminals (υποχρεωτικό)

Με αυτό το JSR θα μπορεί κάποιος να βλέπεις streaming multi-media στην συσκευή του. Το JSR περιλαμβάνει τα πάντα, από έναν ηλεκτρονικό οδηγό προγράμματος, μέσω της αγοράς και DRM, σε μέχρι διαχείριση σύνδεσης και παρουσίαση. Το JSR 272 επίσης προβλέπει την πρόσβαση σε βοηθητικές μετάδοσεις δεδομένων σε όποια ειδική μορφή μπορεί να λάβει η συσκευή.

JSR 280: XML API για Java ME (υποχρεωτικό)

XML είναι η κοινή γλώσσα, για την ανταλλαγή δεδομένων. Είναι ήδη σε χρήση για τις Java ME platforms. Το XML API για την Java ME θα επιτρέψει ευρεία χρήση της XML σε Java ME εφαρμογές για την αποθήκευση και την ανταλλαγή δεδομένων μεταξύ των εφαρμογών, των συσκευές και των απομακρυσμένων υπηρεσιών.

JSR 281: Internet Multimedia Subsystem Services API (υποχρεωτικό)

Με την πρώτη ματιά, Internet Multimedia Subsystem (IMS) ακούγεται απλώς σαν μια quality of service (QoS) υλοποίηση για το Java ME περιβάλλον. Ωστόσο, το QoS είναι μόνο ένα μικρό μέρος της προδιαγραφής. Το IMS είναι μια πλατφόρμα πολυμέσων για packet-oriented δίκτυα πρόσβασης - UMTS, EDGE, GPRS ή W-LAN. Επιτρέπειμία και μόνο-σύνδεση για πρόσβαση σε φωνητικές, πολυμεσικές υπηρεσίες, και υπηρεσίες δεδομένων με ασφαλή τρόπο, κάνοντας χρήση session-control υπηρεσιών (registration, routing, and roaming).

JSR 287: Scalable 2D Vector Graphics API 2.0 για Java ME (υποχρεωτικό)

Το SVG 2 είναι η τελευταία αναθεώρηση του scalable vector 2D graphics για την Java ME platform. Η νέα έκδοση θα προσθέσει υποστήριξη για use cases τα οποία αφορούν τη διασκέδαση, οπτικοποίηση χαρτών, το 2D gaming, και το rendering εγγράφων.

JSR 293: Location API 2.0 (υποχρεωτικό)

Αυτή είναι μια αναθεώρηση τουαρχικού Location API (JSR 179). Προσθέτει υποστήριξη για geocoding, ενσωματώνει στους χάρτες ορόσημα και πλοήγηση, ενώ επίσης, επεκτείνει τις λειτουργίες του Location API 1.0 .

Όλα τα υπόλοιπα JSRs της MSA παραμένουν ίδια και στην MSA2.

[11]

3.5 Το Generic Connection Framework

3.5.1 Εισαγωγή στο Generic Connection Framework

Το Generic Connection Framework (GCF) είχε αρχικά οριστεί να βασίζεται στο Connected Limited Device Configuration (CLDC) 1.0, επειδή τα APIs java.net και java.io της J2SE είχαν θεωρηθεί αρκετά μεγάλα για να χωρέσουν στην περιορισμένη διαθέσιμη μνήμη των κινητών.

Από το 2003 ήδη μπορεί κάποιος να συναντήσει το GCF όχι μόνο σε profiles βασισμένα στο CLDC, όπως το Mobile Information Device Profile (MIDP) και το Information Module Profile (IMP), αλλά επίσης και σε profiles βασισμένα στο Connected Device Configuration (CDC), όπως το Foundation Profile, το Personal Basis Profile και το Personal Profile, ενώ με το JSR 197 και σε πλατφόρμες J2SE. Ακόμη, μπορεί κάποιος να το συναντήσει σε ολοένα και αυξανόμενα προαιρετικά πακέτα, συμπεριλαμβανομένων εκείνων που προσφέρουν υποστήριξη σε Bluetooth και πρόσβαση σε αρχεία

και smart cards.

3.5.2 Μία γενική προσέγγιση στην συνδεσιμότητα

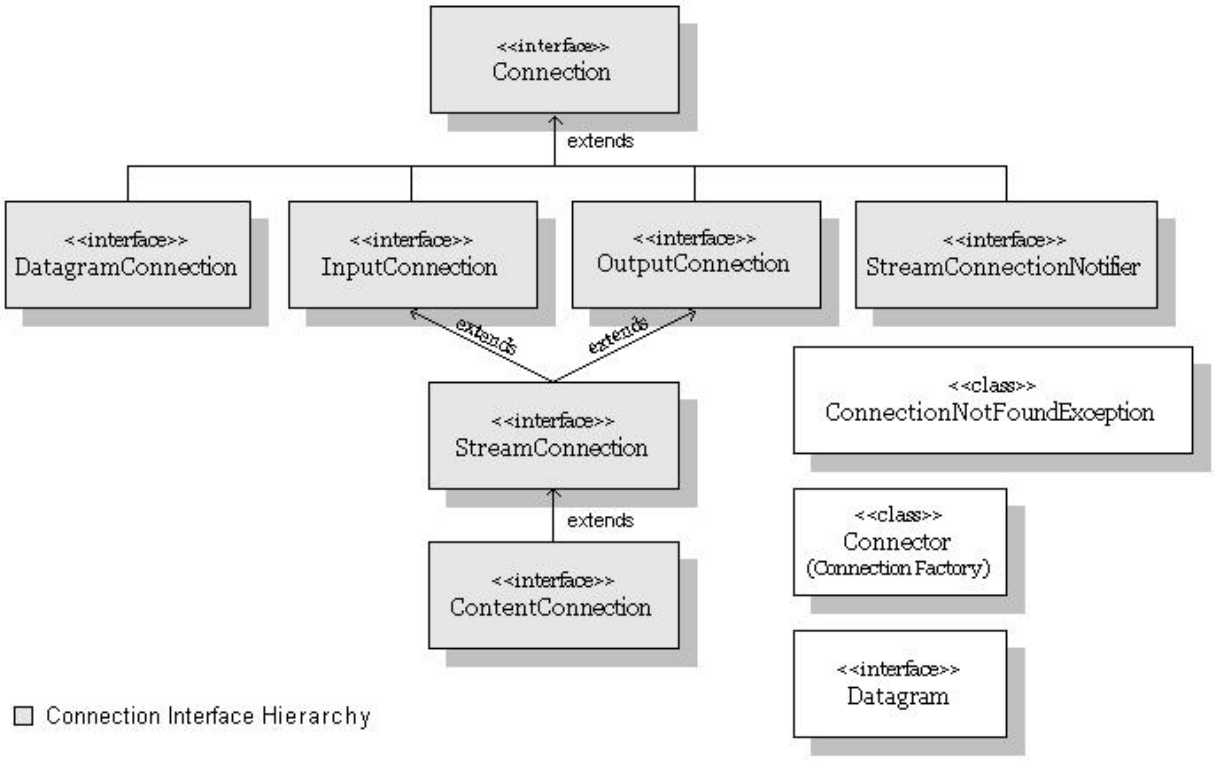
Το GCF είναι μία απλά ιεραρχία από interfaces και classes για να δημιουργήσουν επικοινωνίες όπως HTTP, datagram ή streams.

Όπως υπονοεί και το όνομά του, το GCF παρέχει μία generic προσέγγιση στην συνδεσιμότητα. Είναι generic, επειδή παρέχει μία κοινή ίδρυση API για όλους τύπους βασικούς συνδέσεων. Όπως για συνδέσεις βασισμένες σε πακέτα (data blocks) και βασισμένες σε stream (συνεχόμενες ή σειριακές) input και outpout.

Αυτή η γενίκευσης είναι δυνατή, μέσως της χρήσης:

- Μία ιεραρχίας interface η οποία είναι επεκτάσιμη
- Μίας connection factory
- Standard Uniform Resource Locators (URLs) για να επισημαίνουν τι τύποι συνδέσεων να δημιουργηθούν

Το βασικό CLDC 1.0 GCF φαίνεται στην εικόνα:



Εικόνα 3.5 - Η ιεραρχία του Connection Interface

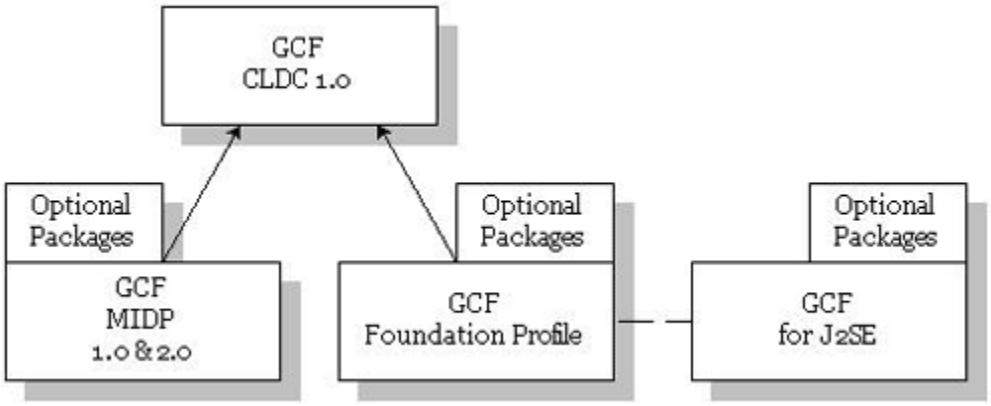
Στην κορυφή της ιεραρχίας του interface είναι το *Connection* interface, το πιο βασικό είδος σύνδεσης. Όλα τα άλλα είδη σύνδεσης επεκτείνουν το *Connection*. Καθώς επεκτείνεται η ιεραρχία, οι συνδέσεις γίνονται πιο πολύπλοκες και λειτουργικές.

Για I/O βασισμένο σε πακέτα, το GCF ορίζει το *DatagramConnection* και για I/O βασισμένο σε stream, υπάρχουν τα *InputConnection, OutputConnection, StreamConnection* και *ContentConnection.* Να σημειωθεί πως το *StreamConnection* επεκτείνει και το *InputConnection* και το *OutputConnection*, δημιουργώντας έτσι μία σύνδεση two-way stream. Στο κάτω μέρος της ιεραρχίας, υπάρχει το *ContentConnection,* ένα ειδικό είδος *StreamConnection* το οποίο ορίζει πληροφορίες σχετικές με το περιεχόμενο όπως το μήκος των δεδομένων, το είδος του περιεχομένου, και η κωδικοποίηση των δεδομένων. Τέλος, το *StreamConnectionNotifier* καθιστά μία εφαρμογή ικανή στο να περιμένει ασύγχρονες εισερχόμενες stream συνδέσεις.

Επί προσθέτως, στην ιεραρχία σύνδεσης, το GCF παρέχει την class *Connector,* η οποία βρίσκεται στο connection factory και την *ConnectionNouFoundException*, η οποία χρησιμοποιείται για να ενδεικνύει πότε ένας τύπος σύνδεσης δεν μπορεί να δημιουργηθεί. Για συνδέσεις βασισμένες σε πακέτα, το GCF ορίζει το interface *Datagram,* το οποίο αναπριστά datagram πακέτα.

Όχι ορισμένα από το GCF, αλλά σχετικά με το αυτό, είναι τα *InputStream, DataInputStream, OutputStream* και *DataOutputStream*, γνωστά στους χρήστες από το *java.io* πακέτο, για συνδέσεις βασισμένες σε streams.

Το GCF είναι τόσο πρακτικό και ευέλικτο, που χρησιμοποιείται παντού στα J2ME profiles και προαιρετικά πακέτα, και από το 2003 και στην J2SE. Η σχέση μεταξύ του CLDC GFC και του GFC για profiles και πλατφόρμες, μπορεί να φανεί στην παρακάτω εικόνα:



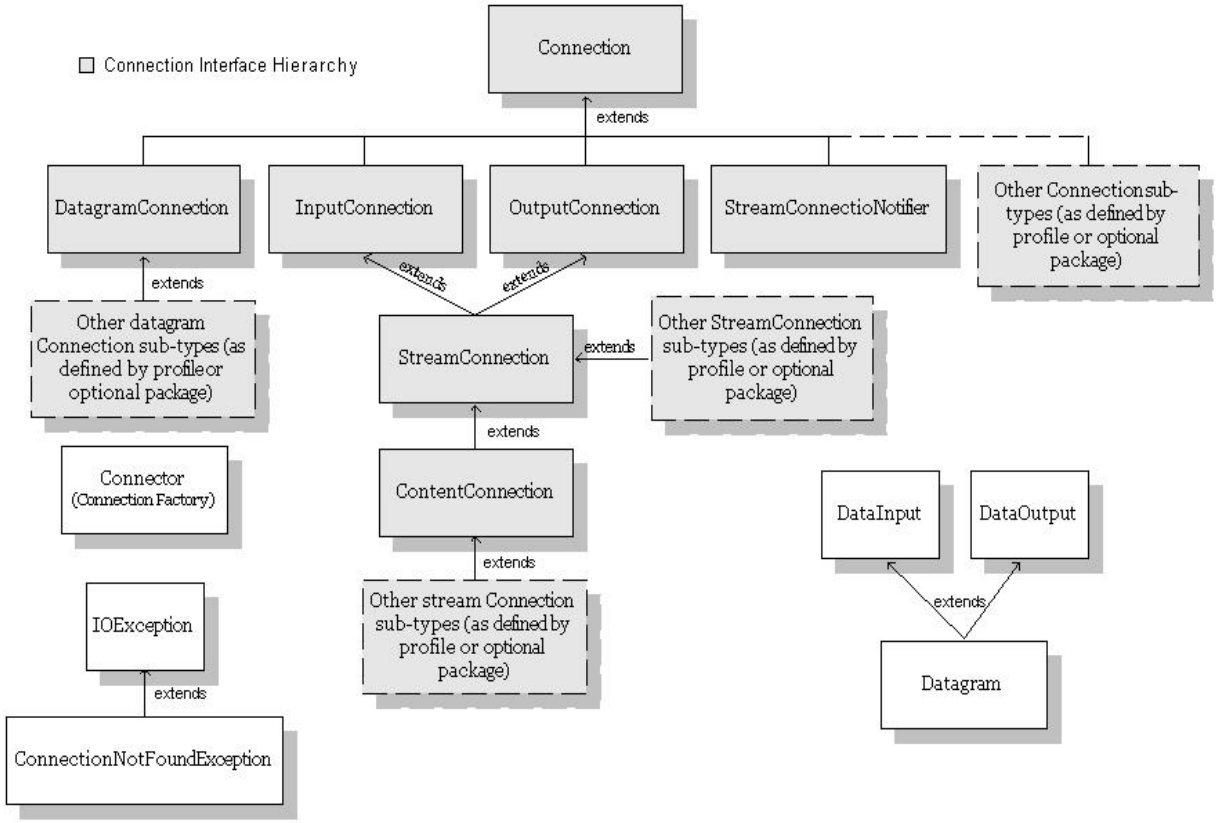
Εικόνα 3.6 - Σχέσεις μεταξύ βασικού GFC και GFC για MIDP, FP και J2SE

3.5.3 Ένα επεκτάσιμο framework για όχι και τόσο generic συνδέσεις

Το GCF το οποίο ορίζεται στο *javax.microedition.io* πακέτο, ορίζει ένα χαμηλού επιπέδου framework μία υψηλού επιπέδουν γενίκευση. Εναπόκεται στα profiles και στα προαιρετικά πακέτα να επεκτείνουν το βασικό GCF και να ορίσουν και να παρέχουν τους χαμηλού επιπέδου τύπους σύνδεσης και το τις υλοποιήσεις του δικτύου και του I/O πρωτοκόλου.

Όταν δούμε από πολύ κοντά μία τυπική σύνδεση και το σχετικό με αυτή πρωτόκολο I/O, βλέπουμε πως δεν είναι generic. Για παράδειγμα, HTTP συνδέσεις έχουν ιδιαιτερότητες οι οποίες ανακλούν their request/response nature, while a Bluetooth connection exposes the dynamic (ad hoc) nature of its protocol, and a socket connection exposes low-level networking methods for timeouts and keep-alive options.

Ευτυχώς, το GCF είναι επεκτάσιμο. Νέοι τύποι σύνδεσεις, οι οποίοι ορίζονται και πρωτυποποιούνται από την Java Community Process (JCP), μπορούν να προστεθούν ορίζοντας έναν νέο υποτύπο Connection και class που να το υποστηρίζει, παρέχοντας μία Connector factory class η οποία υποστηρίζει το νεοορισμένο τύπο σύνδεσης, και ορίζωντας ένα νέο scheme URL το οποίο αναγνωρίζεις το νέο τύπο σύνδεσης. Η παρακάτω εικόνα δείχνει πως ένα συγκεκριμένο profile ή προαιρετικό πακετό, μπορεί να επεκτείνει το GCF:



Εικόνα 3.7 - Επεκτείνοντας το Generic Connection Framework

3.5.4 Ένα πλούσιο σύνολο από τύπους συνδέσεων

Το GCF υποστηρίζει έναν μεγάλο αριθμό τύπων συνδέσεων σε όλα τα profiles και τις πλατφόρμες με έναν σταθερό τρόπο.

Τα URLs παίζουν έναν πολύ σπουδαίο ρόλο στο GCF. Τα URLs περιγράφουν μία τοποθεσία και την μέθοδο πρόσβασης για μία τοποθεσία στο Internet, χρησιμοποιώντας μία ιεραρχική σημειογραφία. Στο GCF τα URLs αναγνωρίζουν τους τύπους σύνδεσης και τα σημεία τέλους.

Η γενική μορφή ενός URL όπως ορίζεται στο RFC 1738 κι αργότερα στο RFC 2396, είναι ως ακολούθως:

scheme://user:password@host:port/url-path;parameters

όπου

- Το scheme ορίζει την μέθοδος πρόσβασης ή το πρωτόκολο όπως FTP ή HTTPS. Στο GCF περιγράφει τον τύπο της σύνδεσης που θα χρησιμοποιηθεί, το οποίο χαρτογραφεί την υποκείμενη σύνδεση ή το I/O πρωτόκολο
- user είναι ένα προαιρετικό user name
- password είναι ένα προαιρετικό password
- host είναι ένα πλήρες όνομα, ή μία IP διεύθυνση του host όπου βρίσκεται το resource
- port είναι ένα προαιρετικό port προς χρήση. Η διερμήνευσή του, εξαρτάται από το scheme
- url-path είναι το path προς το resource. Η μορφή και διερμήνευσή του, εξαρτόνται από το scheme. Το url-path μπορεί να ορίζει προαιρετικές παραμέτρους

Ειδικής σημασίας είναι το scheme το οποίο στο GCF περιγράφει το είδος της σύνδεσης που θα δημιουργηθεί. Ο παρακάτω πίνακας συνοψίζει μερικά από τα GCF URL schemes και τύπους συνδέσεων, όπως ορίζονται από την JCP:

Πίνακας 3.1 - GCF URL Schemes και τύποι συνδέσεων

URL Scheme	Συνδεσιμότητα	Τύπος Σύνδεσης GFC	Ορίζεται από
bt12cap	Bluetooth	L2CAPConnection	JSR 82 Προαιρετική υποστήριξη
datagram	Datagram	DatagramConnection	Όλα τα profiles βασισμένα στα CLDC και CDC, όπως το MIDP, Foundation. JSR 197, J2SE. Υποστήριξη προαιρετική
file	File Access	FileConnection, InputConnection	JSR 75 Προαιρετική υποστήριξη
http	HyperText Transfer	HttpConnection	MIDP 1.0, MIDP 2.0, Foundation Profile, J2SE (JSR 197).

URL Scheme	Συνδεσιμότητα	Τύπος Σύνδεσης GFC	Ορίζεται από
	Protocol		Προαιρετική υποστήριξη
https	Secure Http	HttpsConnecton	MIDP 2.0. Προαιρετική υποστήριξη.
comm	Serial I/O	CommConnection	MIDP 2.0. Προαιρετική υποστήριξη.
sms mms cbs	Short Messaging Service Multimedia Messaging Service Cell Broadcast SMS	MessageConnection	JSR 120, JSR 205. Προαιρετική υποστήριξη
apdu jcrmi	Security Element	APDUConnection, JavaCardRMIConnection	JSR 177. Προαιρετική υποστήριξη
socket serversocket	Socket	SocketConnection, ServerSocketConnection	JSR118 (MIDP 2.0). Προαιρετική υποστήριξη
datagram	UDP Datagram	UDPDatagramConnection	JSR118 (MIDP 2.0). Προαιρετική υποστήριξη

Να σημειωθεί πως οι περισσότεροι τύποι συνδέσεων είναι προαιρετικοί. Οι κατασκευαστές των συσκευών δεν είναι υποχρεωμένοι να τους υποστηρίξουν και σαν αποτέλεσμα, ορίζονται από ένα προαιρετικό πακέτο. Πριν γίνει η χρήση κάποιου API θα πρέπει να βεβαιωθεί ο προγραμματιστής ότι χρησιμοποιεί το κατάλληλο JSR.

Ο παρακάτω πίνακας συνοψίζει του GCF τύπους συνδέσεων κατά profile:

Πίνακας 3.2 - Περίληψη του πυρήνα των GCF τύπων συνδέσεων, με βάση το Profile

GCF Interfaces, Classes, Exceptions	Πρέπει να υποστηρίζουν οι κατασκευαστές;	CLDC 1.0, 1.1	MIDP 1.0	MIDP 2.0	Foundation και σχετικά Profiles	GCF για J2SE (JSR 197)
CommConnection	Όχι			X		
Connection (βασική σύνδεση)	Ναι	X	X	X	X	X
Content Connection	Ναι	X	X	X	X	X
Datagram Connection	Όχι	X	X	X	X	X
HttpConnection	Ναι		X	X	X	X

GCF Interfaces, Classes, Exceptions	Πρέπει να υποστηρίζουν οι κατασκευαστές;	CLDC 1.0, 1.1	MIDP 1.0	MIDP 2.0	Foundation και σχετικά Profiles	GCF για J2SE (JSR 197)
HttpsConnection	Ναι			X		
InputConnection	Ναι	X	X	X	X*	X*
OutputConnection	Ναι	X	X	X	X	X
SecureConnection	Όχι			X		
ServerSocket Connection	Όχι			X		
SocketConnection	Όχι			X		
StreamConnection	Ναι	X	X	X	X	X
StreamConnection Notifier	Ναι	X	X	X	X	X
UDPDatagram Connection	Όχι			X		

* Ένα URL scheme αρχείου: επιστρέφει μία InputConnection

Το MIDP 2.0 GCF παρέχει τους περισσότερους τύπους σύνδεσης, ενώ τα άλλα profiles και η J2SE υποστηρίζουν μόνο την HttpConnection, εκτός από τους βασικούς τύπους CLDC GCF.

3.5.5 Χρησιμοποιώντας το Generic Connection Framework

Η χρήση του GCF είναι πολύ απλή. Η δημιουργία μίας σύνδεσης απαιτεί την class Connector factory και ένα URL. Για να κλείσει, γίνεται χρήση του αντικειμένου του υποτύπου της Connection που είχε δημιουργηθεί. Η χρήση μίας σύνδεσης μπορεί να είναι από απλή έως πολύπλοκη, ανάλογα με τον τύπο σύνδεσης με τον οποίο θα πραγματοποιηθεί.

Ανοίγοντας μία σύνδεση με την Connector Class

Η class Connector είναι το connection factory. Δημιουργεί και ανοίγει ένα instance του κατάλληλου υποτύπου της Connection, βασιζόμενη στο URL που ορίζεται, εάν αυτός ο τύπος της σύνδεης υποστηρίζεται.

Για την δημιουργία μίας σύνδεσης, χρησιμοποιείται η μέθοδος *Connection.open()* όπως φαίνεται στο παρακάτω κομμάτι κώδικα:

```
...
String url = "socket://www.j2medeveloper.com:80";
...
SocketConnection c = null;
InputStream s = null;
try {
    c = (SocketConnection)Connector.open(url);
```

```
    s = c.openInputStream();
    ...
    // Read from the input stream, handle input data.
    ...
} catch (ConnectionNotFoundException cne) {
    // Connection specified in URL can't be created.
    // Handle exception, throw exception or return error.
} catch (IllegalArgumentException iae) {
    // One of the arguments is in error.  In this example, the
    // only argument to open is URL, so the only expected
    // exceptions are ConnectionNotFoundException or IOException.
    // Handle exception, throw exception or return error.
} catch (IOException ioe) {
    // Handle exception, throw exception or return error.
} finally {
    try {
        if (s != null) s.close();
        if (c != null) c.close();
    } catch (Exception e) {
        // Handle exception, throw exception or return error.
    }
}
...
```

Η class Connector ορίζει τρεις μεθόδους open()

- open(String url)
- open(String url, int mode)
- open(String url, int mode, boolean timeouts)

όπου

- url είναι το URL το οποίο περιγράφει τον τύπο της σύνδεσης
- mode είναι τι είδος σύνδεσης θα χρησιμοποιηθεί – READ, WRITE ή READ_WRITE (default)
- timeouts είναι η Boolean μεταβλητή που δείχνει ότι ο caller θέλει να ειδοποιηθεί για timeout exceptions (InterruptedIOException), εάν η υποκείμενη σύνδεση υλοποιεί την ειδοποίηση (Το default είναι false)

Η απλή μορφή της μεθόδου open(), η open(String url) είναι η πιο διαδεδομένη στην χρήση:

```
...
// Create a SocketConnection and InputStream
String url = "socket://www.j2medeveloper.com:80";
c = (SocketConnection)Connector.open(url);
s = c.openInputStream();
...
```

Η class Connector διαθέτει αρκετά βολικές μεθόδους για την δημιουργία streams διαφόρων τύπων:

- static DataInputStream openDataInputStream(String name)
- static DataOutputStream openDataOutputStream(String name)
- static InputStream openInputStream(String name)
- static OutputStream openOutputStream(String name)

Επειδή αυτές οι μέθοδοι είναι static, μπορεί να γίνει χρήση τους όπως φαίνεται παρακάτω:

```
...
// Create an InputStream
String url = "socket://www.j2medeveloper.com:80";
s = (InputConnection)Connector.openInputStream(url);
...
```

Καλό θα ήταν αυτή η μέθοδος ανοίγματος των streams να αποφεύγεται, επειδή δεν γίνεται αναφορά στην σύνδεση αυτή καθεαυτή, αλλά κι επειδή δεν θα έχει κάποιος πρόσβαση μεθόδους σχετικές με την σύνδεση ή σε χαρακτηριστικά. Για παράδειγμα, εάν κάποιος χρησιμοποιήσει ένα URL στην openInputStream, το οποίο ορίζει http, δεν θα μπορεί κάποιος να ΄χει πρόσβαση σε καμία από της μεθόδους της HttpConnection, ώστε να
χειριστεί τα HTTP headers.

Παρακάτω παρατείθενται ορισμένα πραδείγματα δημιουργίας διαφόρων τύπων σύνδεσης:

Δημιουργώντας μία `SocketConnection`:

```
...
String url = "socket://www.j2medeveloper.com:80";
SocketConnection c = (SocketConnection)Connector.open(url);
...
```

Δημιουργώντας μία `HttpConnection`:

```
...
String url = "http://www.j2medeveloper:80/com/myServlet";
HttpConnection c = (HttpConnection)Connector.open(url);
...
```

Δημιουργώντας μία `InputConnection` (Foundation Profile and J2SE):

```
...
String url = "file:///myResourceFile.res";
InputConnection c = (InputConnection)Connector.open(url);
...
```

Δημιουργώντας μία `FileConnection` (JSR 75):

```
...
String url = "file:///myResourceFile.res";
FileConnection c = (FileConnection)Connector.open(url);
...
```

Δημιουργώντας μία `DatagramConnection`:

```
String url = "datagram://www.j2medeveloper.com:7001";
UDPDatagramConnection c =
(UDPDatagramConnection)Connector.open(url);
```

Κλείνοντας μία σύνδεση χρησιμοποιώντας το Connection Interface

Το Connection interface ορίζει τον πιο βασικό τύπο σύνδεσης. Η μόνης του μέθοδος είναι η *close()* . Υποθέτωντας πως η σύνδεση που έχουμε δημιουργήσει ονομάζετια conn, το παρακάτω κομμάτι κώδικα δείχνει πως πρέπει να χρησιμοποιείται αυτή η μέθοδος:

```
try {
    // Here close any open streans
    conn.close();
} catch (IOException ioe) {
    // Handle the exception.
    // Throw the exception, ignore it, or return an error.
}
```

[19]

3.6 Ανακεφαλαίωση

Στο κεφάλαιο αυτό είδαμε τις δυνατότητες των MSA, MSA2, JTWI και Generic Connection Framework. Στο επόμενο που ακολουθεί θα δούμε τα πάντα για τα Configurations, τα Profiles και τα Optional Packages, αυτά τα οποία συνιστούν στην ουσία την J2ME.

Κεφάλαιο 4ο

4.1 Εισαγωγή

Σε αυτό το κεφάλαιο θα δούμε τα "δομικά" συστατικά της J2ME. Τα Configurations, τα Profiles και τα Optional Packages. Όλα τους στην ουσία είναι βιβλιοθήκες, με ξεχωριστή σημασία και βαρύτητα όμως το καθένα. Από μόνα τους έχουν λίγες δυνατότητες, ο συνδυασμός τους όμως μπορεί να κάνει θαύματα. Το μόνο που μένει είναι μέσα από τις παρακάτω σελίδες, να τα κατανοήσει ο χρήστης τον ρόλο και την λειτουργία τους.

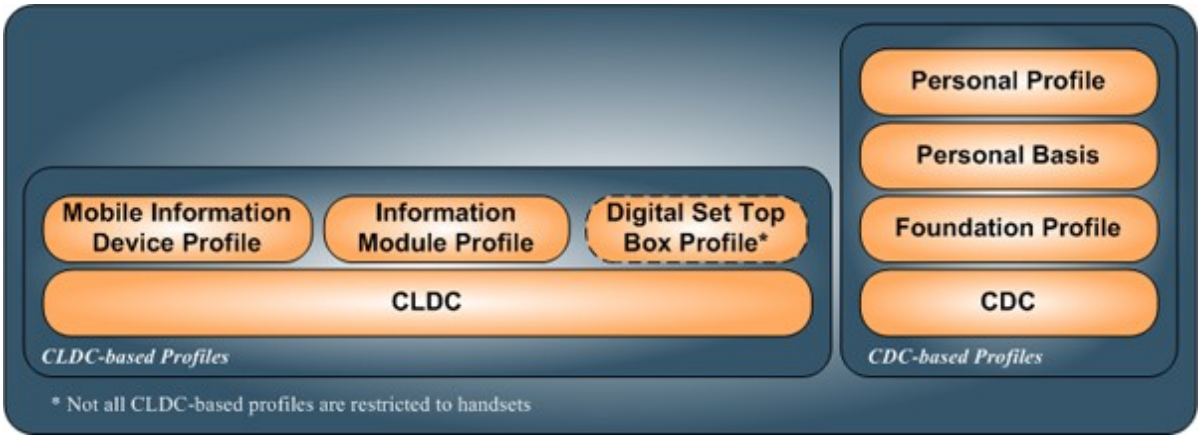
4.2 Εισαγωγή στα Profiles και στα Configurations

4.2.1 Βασικά μέρη της J2ME

Ενώ οι καταναλωτικές συσκευές όπως τα κινητά τηλέφωνα, οι pagers, οι personal organizers και τα set-top boxes, έχουν πολλά κοινά στοιχεία, είναι αρκετά διαφορετικές ως προς τη μορφή, τη λειτουργία και τις δυνατότητες. Για την αντιμετώπιση αυτής της ποικιλίας, ένα απαραίτητο για την J2ME όχι μόνο να είναι μικρή, αλλά και προσαρμόσιμη. Η αρχιτεκτονική της J2ME είναι modular και κλιμακούμενη, ώστε να μπορεί να στηρίξει τα ευέλικτα είδη χρήσης τα οποία απαιτούνται από τον καταναλωτή. Για να επιτευχθεί αυτό, η J2ME προβλέπει μια σειρά από virtual machine τεχνολογίες , κάθε μία βελτιστοποιημένη για διαφορετικά είδη επεξεργαστή και μνήμης. Για χαμηλού επιπέδου, περιορισμένων πόρων προϊόντα, η J2ME υποστηρίζει μικρά configurations της Java Virtual Machine και των Java APIs τα οποία ενσωματώνουν μόνο τις βασικές δυνατότητες για κάθε είδος συσκευής. Καθώς οι κατασκευαστές αναπτύσσουν νέα χαρακτηριστικά στις συσκευές τους, ή οι πάροχοι υπηρεσιών αναπτύσσουν νέες και συναρπαστικές εφαρμογές, αυτά τα μικρά configurations μπορούν να επεκταθούν με επιπρόσθεταAPIs ή με τη συμπλήρωση των πλουσιότερων Java Virtual Machine χαρακτηριστικών. Για την υποστήριξη αυτού του είδους της προσαρμογής και της επεκτασιμότητας, δύο βασικές έννοιες ορίζονται από την J2ME:

- Configuration: Ένα J2ME configuration καθορίζει την ελάχιστην platfom για μία "horizontal" κατηγορία ή ομάδα συσκευών με παρόμοιες απαιτήσεις σχετικά με το σύνολο της μνήμης και την επεξεργαστική ισχύ. Ένα configuration ορίζει τα χαρακτηριστικά και τις ελάχιστες βιβλιοθήκες από classes της Java και της virtual machine, τα οποία θα πρέπει ένας κατασκευαστής να περιμένει να είναι διαθέσιμα σε όλες τις συσκευές της ίδιας κατηγορίας.

- Profile: Ένα J2ME profile βασίζεται (και συνεπώς, επεκτείνει) ένα configuration. Ένα profile απευθύνει συγκεκριμένες απαιτήσεις σε ένα "vertical" τμήμα της αγοράς ή οικογενείας συσκευών. Ο κύριος στόχος ενός profile είναι να εξασφαλίσει την διαλειτουργικότητα μέσα σε μία οικογένεια συσκευών ή τομέα, ορίζοντας μία πρότυπη Java platform για εκείνη την αγορά. Το Profile συνήθως sπεριλαμβάνει βιβλιοθήκες, οι οποίες είναι πιο συγκεκριμένες σε έναν τομέα, από ότι οι βιβλιοθήκες από classes οι οποίες παρέχονται από ένα configuration.



Εικόνα 4.1 - Τα Configurations και Profiles της J2ME platform

[21]

4.2.2 J2ME Profiles

Η φορητότητα της εφαρμογής αποτελεί βασικό όφελος της Java τεχνολογίας. Η δυνατότητα μεταφοράς αποτελεί επίσης ένα κρίσιμο στοιχείο της J2ME για τις καταναλωτικές συσκευές. Ωστόσο, οι ανάγκες για την φορητότητα μιας εφαρμογής διαφέρει για desktop και για κινητές συσκευές. Στην πλειοψηδία των περιπτώσεων οι καταναλωτικές συσκευές έχουν ουσιαστικές διαφορές στο μέγεθος της μνήμης, στην δικτύωση και στις δυνατότητες του interface, γεγονός που καθιστά πολύ δύσκολη την υποστήριξη προς όλες τις συσκευές με μία μόνο λύση.

Σε γενικές γραμμές, η αγορά συσκευών δεν είναι τόσο ομοιογενής ενώ οι τελικοί χρήστες αναμένουν ή να απαιτούν την καθολική εφαρμογή της φορητότητας. Όμως οι εφαρμογές θα πρέπει να είναι πλήρως φορτητές ανάμεσα σε συκευές του ιδίου είδους. Για παράδειγμα, σκεφτείτε τις παρακάτω είδη των καταναλωτικών προϊόντων:

- Κινητά τηλέφωνα
- Πλυντήρια ρούχων
- Ηλεκτρονικά παιχνίδια τα οποία επικοινωνούν μεταξύ τους

Είναι σαφές ότι κάθε ένα από αυτά αντιπροσωπεύει ένα διαφορετικό τμήμα της αγοράς ή οικογένεια συσκευών ή τομέα εφαρμογών. Ως εκ τούτου, οι καταναλωτές θα ανέμεναν οι χρήσιμες εφαρμογές για να είναι φορητές μεταξύ συσκευών της ιδίας οικογένειας. Για παράδειγμα θα μπορούσε κάποιος να πει τα παρακάτω:

- Θα περίμενα η αγαπημένη μου εφαρμογή να λειτουργεί το ίδιο σε όλα μου τα κινητά, αν και είναι από διαφορετικούς κατασκευαστές.
- Αν έβρισκα ένα πρόγραμμα πλυντηρίου το οποίο παίζει στο παλιό μου πλυντήριο, αλλά όχι στο καινούριο, θα ήταν πολύ ενοχλητικό.
- Το πάρτυ γενεθλίων του παιδιού μου, θα είναι λιγότερο ευχάριστο, αν το νέο ρομπότ δεν παίζει

με το ηλεκτρονικό αρκουδάκι.

Από την άλλη, κανείς καταναλωτή δεν περιμένει οι λειτουργίες των παραπάνω λεγομένων να αλληλεπιδράσουν μεταξύ τους. Με άλλα λόγια, η φορητότητα μίας εφαρμογής σε διαφορετικές οικογένειες συσκευών δεν είναι αναγκαστικά τόσο πολύ σημαντική.

- Ένα profile ορίζει μία Java platform για ένα συγκεκριμένο τμήμα της αγοράς. Τα profiles μπορούν να εξυπηρετήσουν δύο απαιτήσεις φορητότητας: Ένα profile προσφέρει ένα ολοκληρωμένο toolkit για την υλοποίηση εφαρμογών για ένα συγκεκριμένο είδος συσκευής, όπως ένα pager, ένα set-top box, ένα κινητό τηλέφωνο, ένα πλυντήριο ή ένα διαδραστικό ηλεκτρονικό παιχνίδι.

- Ένα profile επίσης μπορεί να δημιουργηθεί για να υποστηρίξει μια σημαντική, συνεκτική ομάδα από εφαρμογές οι οποίες θα μπορούσαν να φιλοξενούνται σε διάφορες κατηγορίες προϊόντων. Για παράδειγμα, ενώ οι διαφορές μεταξύ των set-top boxes, των pagers, των κινητών τηλεφώνων, και των συσκευών πλυσίματος είναι αρκετά σημαντικές ώστε να δικαιολογούν τη δημιουργία ενός ξεχωριστού profile για το κάθε ένα, θα μπορούσε να είναι χρήσιμο για ορισμένα είδη διαχείρισης προσωπικών πληροφοριών, ή διαχείρισης τραπεζικών συνδιαλλαγών να μπορούν να είναι φορητά σε αυτές τις συσκευές. Αυτό θα μπορούσε να επιτευχθεί με τη δημιουργία ενός ξεχωριστού profile για αυτού τους είδους των απαιτήσεων, καθώς και διασφαλίζοντας ότι αυτό το νέο profile θα μπορεί εύκολα να υποστηριχθεί αποτελεσματικά σε κάθε μία από τις συσκευές τις οποίες στοχεύει η J2ME με το συγκεκριμένο profile.

Είναι δυνατόν για μία συσκευή να υποστηρίξει αρκετά profiles. Ορισμένα από αυτά τα profiles θα βασίζονται στην συσκευή, ενώ άλλα στην εφαρμογή. Εφαρμογές οι οποίες είναι γραμμένες για ένα συγκεκριμένο profile θα πρέπει να χρησιμοποιούν μόνο τα χαρακτηριστικά που ορίζονται από το εν λόγω profile. Οι κατασκευαστές επιλέγουν ποια profiles θα υλοποιήσουν στις συσκευές, όμως θα πρέπει να υλοποιήσουν όλα τα χαρακτηριστικά του profile που επιλέχθηκε. Έτσι μία εφαρμογή θα τρέχει σε όλες τις συσκευές οι οποίες υποστηρίζουν το ίδιο profile με την εφαρμογή.

Με απλά λόγια, το profile είναι μια σύμβαση μεταξύ μίας εφαρμογής και ενός J2ME "vertical" τμήματος της αγοράς. Όλες οι συσκευές στο ίδιο τμήμα της αγοράς συμφωνούν να υλοποιήσουν όλα τα χαρακτηριστικά τα οποία ορίζονται στο profile και η εφαρμογή κάνει χρήση μόνο εκείνα τα χαρακτηριστικά που ορίζονται στο profile. Έτσι, επιτυγχάνεται η φορητότητα μεταξύ των εφαρμογών και των συσκευών που εξυπηρετούνται από το εν λόγω profile. Νέα συσκευές μπορούν να επωφεληθούν από την μεγάλη ποικιλία εφαρμογών. Το πιο σημαντικό όμως είναι, πως και οι νέες εφαρμογές μπορούν να χρησιμοποποιήσουν τις ήδη υπάρχουσες συσκευές.

Σε επίπεδο υλοποίησης, ένα profile ορίζεται απλά ως μια συλλογή από Java APIs και βιβλιοθήκες από classes, τα οποία πατάνε πάνω σε ένα configuration προσφέροντας περισσότερες δυνατότητες.

4.2.3 J2ME Configurations

Στην J2ME, μία εφαρμογή είναι γραμμένη για ένα συγκεκριμένο profile, καθώς και ένα profile βασίζεται, ή επεκτείνει ένα συγκεκριμένο configuration. Έτσι, όλα τα χαρακτηριστικά ενός configuration αυτομάτως συμπεριλαμβάνονται και στο profile και μπορούν να χρησιμοποιηθούν από εφαρμογές γραμμένες για εκείνο το profile.

Ένα configuration ορίζει μια Java platform για μια "horizontal" κατηγορία ή ομάδα συσκευών με παρόμοιες απαιτήσεις οι οποίες αφορούν το συνολικό ποσό της μνήμης και τις ικανότητες του υλικού. Πιο συγκεκριμένα, ένα configuration:

- καθορίζει τα χαρακτηριστικά της Java τα οποία υποστηρίζονται
- καθορίζει τα χαρακτηριστικά Java Virtual Machine τα οποία υποστηρίζονται
- καθορίζει τις βασικές βιβλιοθήκες και τα APIs τα οποία υποστηρίζονται

Η J2ME είναι σχεδιασμένη ώστε να μπορεί να αναπτυχθεί σε περισσότερα από ένα configuration.

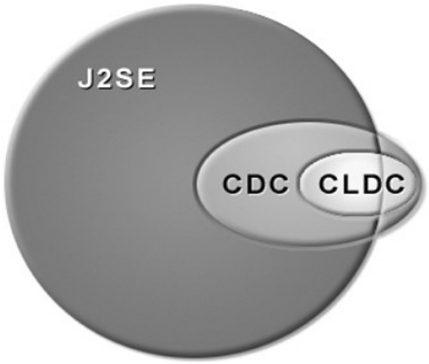
Κάθε configuration καθορίζει τα Java Virtual Machine χαρακτηριστικά και ένα σύνολο από APIs τα οποία το profile το οποίο τα υλοποιεί (και κατά συνέπεια και οι εφαρμογές που χρησιμοποιούν το profile) μπορεί να θεωρήσει πως είναι παρόντα σε όλες τις συσκευές. Με πολύ απλά λόγια, ένα configuration είναι είναι μία σύμβαση μεταξύ ενός profile και της JVM μίας συσκευής. Οι virtual machines όλων των συσκευών στην αγορά συμφωνούν να εφαρμόσουν όλα τα χαρακτηριστικά που ορίζονται στο configuration. Και τα profiles συμφωνούν να χρησιμοποιήσουν μόνο τα χαρακτηριστικά τα οποία καθορίζονται στο configuration. Έτσι, επιτυγχάνεται η φορητότητα μεταξύ των profiles και των συσκευών οι οποίες χρησιμοποιοιούν το συγκρεκριμένο configuration. Οι νέες συσκευές μπορούν να επωφεληθούν από τα υφιστάμενα profiles. Και νέα profiles μπορούν να εγκατασταθούν στις ήδη υπάρχοντες συσκευές.

Στο παράδειγμά μας παραπάνω, καθένα από τα τρία profiles (για κινητά τηλέφωνα, πλυντήρια, και παιχνίδια τα οποία αλληλεπιδρούν μεταξύ τους) είναι πολύ πιθανόν να κατασκευαστούν με βάση το ίδιο configuration, το CLDC. Αυτό το configuration παρέχει το σύνολο των βασικών λειτουργειών οι οποίες είναι απαραίτητες για την εξυπηρέτηση των αναγκών του κάθε profile.

Για να αποφευχθεί η κατακερματισμός, υπάρχει ένας πολύ περιορισμένος αριθμός J2ME configurations. Τα δύο πιο σημαντικά configurations είναι:

- **Connected Limited Device Configuration (CLDC)** Η αγορά η οποία αποτελείται από προσωπικές κινητές συσκευές που συνδέονται σε δίκτυο, εξυπηρετείται από το CLDC. Το configuration αυτό περιέχει μερικές classes, οι οποίες δεν είναι από τα J2SE APIs, αλλά είναι σχεδιασμένες για να ανταποκρίνεται στις ανάγκες των συσκευών με μικρή μνήμη.

- **Connected Device Configuration (CDC)** Η αγορά η οποία αποτελείται από διαμοιραζόμενες συσκευές που συνδέονται στο δίκτυο, εξυπηρετούνται από το Connected Device Configuration (CDC). Για να διασφαλιστεί η συμβατότητα μεταξύ τους, το CDC είναι ένα υπερσύνολο του CLDC.



Εικόνα 4.2 - Οι σχέσεις μεταξύ CDC, CLDC και J2SE

Η παραπάνω εικόνα απεικονίζει τη σχέση μεταξύ CLDC, CDC και Java 2 Standard Edition (J2SE). Όπως φαίνεται στο σχήμα, η πλειοψηφία των λειτουργιών στο CLDC και στο CDC έχουν κληρονομηθεί από την J2SE platform. Επι προσθέτως, τα CLDC και CDC μπορούν να εισάγουν μια σειρά από χαρακτηριστικά, τα οποία δεν προέρχονται από την J2SE, αλλά είναι ειδικά σχεδιασμένα για να ανταποκρίνεται στις ανάγκες των συσκευών με μικρή μνήμη. Ο πιο σημαντικός λόγος για επίπεδο του configuration στην J2ME είναι ότι τα configurations οι JVMseίναι πολύ στενά συνδεδεμένα και πολύπλοκα κομμάτια λογισμικού. Μικρές διαφορές στο configuration της προδιαγραφής μπορεί να απαιτήσει από ένα μεγάλο αριθμό των τροποποιήσεων του εσωτερικού σχεδιασμού της Java Virtual Machine, η οποία θα είναι πολύ δαπανηρή και χρονοβόρα διαδικασία.

Έχοντας έναν μικρό αριθμό configurations σημαίνει ότι ένας σχετικά μικρός αριθμός από υλοποιήσεις της JVM μπορεί να εξυπηρετήσει τις ανάγκες των τη οποία παρέχεται από την J2ME είναι πολύ σημαντική για την επιτυχία και τη σχέση κόστους-αποτελεσματικότητας των συσκευών στην βιομηχανία.

[35]

4.3 Configurations

4.3.1 CLDC

4.3.1.1 Εισαγωγή στο CLDC

To Connected Limited Device Configuration (CLDC) είναι ένα θεμελιώδες κομάτι της αρχιτεκτονικής της Java 2 Platform, Micro Edition (J2ME). Ορίζει ένα βασικό σύνολο από interfaces προγραμματισμού εφαρμογών και μία virtual machine για περιορισμένες από πόρους συσκευές, όπως τα κινητά τηλέφωνα.

Το CLDC καλύπετει τις συσκευές οι οποίες έχουν λιγότερο από 512 KB μνήμης για την χρήση της JVM και καταργεί το JNI, καθορισμένους από τον χρήστη class loaders, και κάποια αντικείμενα για το class verification, ώστε να μπορέσει να ανταποκριθεί στις ανάγκες της μικρής μνήμης. Η JVM που χρησιμοποιεί ονομάζεται KVM όπου το K δείχνει πως η μνήμη είναι τάξη μεγέθους KB. Τα JSRs για το CLDC είναι τα εξής:

- **CLDC 1.0:** Το Connected Limited Device Configuration 1.0 (ή αλλιώς JSR 30), στοχεύει σε συσκευές που έχουν στην κατοχή τους από 128 KB έως 512 KB μνήμης (256 KB ROM / 256 KB RAM ή λιγότερο) ενώ παράλληλα έχουν χαμηλή κατανάλωση μπαταρίας, συνδέονται σε κάποιο δίκτυο κι έχουν user interface.

- **CLDC 1.1:** Το Connected Limited Device Configuration 1.1 (ή αλλιώς JSR 139), στοχεύει σε συσκευές που έχουν στην κατοχή τους από 160 KB μνήμης και πάνω, ενώ η επεξεργαστική τους ταχύτητας κειμένεται από 8 - 32 MHz, κάνουν ό,τι κάνουν και οι συσκευές οι οποίες υποστηρίζονται από το CLDC 1.0, αλλά πλέον τους δίνεται η δυναότητα για να κάνουν πράξει κινητής υποσδιαστολής.

Η CLDC προδιαγραφή ορίζει τρία πράγματα:

- Τις ικανότητες της JVM, η οποία δεν είναι πλήρους χαρακτηριστικών VM
- Ένα μικρό υποσύνολο από J2SE 1.3 classes
- Ένα νέο σύνολο από APIs για I/O, το οποίο καλείται Generic Connection Framework (GCF)

Είναι σημαντικό κάποιος να καταλάβει τι δεν ορίζει το CLDC. Δεν ορίζει κανένα API σχετικό με user interface. Το CLDC δεν ορίζει πως οι εφαρμογές φορτώνονται σε μία συσκευή ή πως ενεργοποιούνται. Τέτοια πράγματα, ορίζονται από τα profiles, τα οποία έχουν ως βάση το CLDC.

Στόχοι

Ο στόχος της προδιαγραφής του CLDC είναι να προτοτυποποιήσει μία υψηλά φορητή, αλλά και με την χρήση λίγων πόρων, πλατφόρμα ανάπτυξης για χαμηλών πόρων σύσκευες οι οποίες συνδέονται και σε δίκτυο. Έχει αναπτυχθεί μέσα στην Java Community Process (JCP), η οποία αποτελείται από εκατοντάδες μέλη, μέσα στα οποία υπάρχουν πάροχοι, κατασκευαστές και εταιρείες ανάπτυξης λογισμικού. Το CLDC έχει κατασκευαστεί με τους παρακάτω στόχους:

- Να μειωθούν οι απαιτήσεις σε επίπεδα κατάλληλα για την μαζική ανάπτυξη
- Διευκόλυνση της μεταφερσιμότητας μετατρέποντας τα πρωτογενή λειτουργικά συστήματα σε πρότυπα APIs
- Επέκτασης της λειτουργικότητας της συσκευής επιτρέποντας δυναμική προσθήκη των εφαρμογών στην συσκευή.

Συσκευές Στόχοι

Το CLDC είναι σχεδιασμένο να φέρει τα πολλά πλεονεκτήματα της Java πλατφόρμα σε συνδεόμενες

στο δίκτυο συσκευές, οι οποίες όμως έχουν περιορισμένη επεξεργαστική ισχύ, μνήμη και ικανότητα για την χρήση γραφικών, όπως τα κινητά τηλέφωνα, οι pagers, φθηνά προσωπικά organizer. Επι προσθέτως, το CLDC μπορεί επίσης να αναπτυχθεί σε εφαρμογές στο σπίτι. TV set-top boxes, and point-of-sale terminals. Οι συσκευές στις οποίες στοχεύει το CLDC έχουν τα παρακάτω χαρακτηριστικά:

- Ένα 16-bit ή 32-bit επεξεργαστεί με ένα ρολόι στα 16MHz ή υψηλότερα
- Τουλάχιστον 160 KB μη πτητικής μνήμης η οποία κατανέμεται στις CLDC βιβλιοθήκες και στην εικονική μηχανή.
- Τουλάχιστον 192 KB συνολική μνήμης διαθέσιμη για την Java πλατφόρμα
- Χαμηλή κατανάλωση ενέργειας, ενώ συχνά λειτουγεί με μπαταρία
- Σύνδεση σε κάποιου είδους δίκτυο, συχνά με ασύρματη, διακοπτόμενη σύνδεση και περιορισμένο εύρος

[31]

Πίνακας 4.1 - Java Packages στο CLDC

Όνομα	Περιγραφή
`java.lang`	CLDC υποσύνολο της Java
`java.lang.ref` (CLDC 1.1 μόνο)	CLDC υποσύνολο για την υποστήριξη των weak references
`java.util`	CLDC υποσύνολο των Java SE utility classes
`java.io`	CLDC υποσύνολο του system I/O μέσω data streams
`javax.microedition.io`	CLDC υποστήριξη δικτύου βασισμένο στο Generic Connection Framework

[21]

Μόνο μερικές επελεγμένες classes περιλαμβάνονται: για παράδειγμα, περιέχονται οι java.util.Vector, και η java.util.Hashtable classes, αλλά όχι οι classes οι οποίες αφορούν τα collections. Το μεγαλύτερο πακέτο είναι το java.lang, το οποίο ορίζει βασικές classes για την δημιουργία οποιασδήποτε Java εφαρμογής. Το java.io περιέχει μόνο abstract και βασισμένες στη μνήμη classes και interfaces όπως τα java.io.DataInput ή το java.io.ByteArrayInputStream, ενώ το java.util δεν περιέχει παρά ελάχιστες classes [7].

4.3.1.2 Διαφορές μεταξύ CLDC 1.0 και CLDC 1.1

Η λίστα παρακάτω συνοψίζει τις κύριες διαφορές μεταξύ του CLDC 1.1 και CLDC 1.0.

- Υποστήριξη δεκαδικών αριθμών

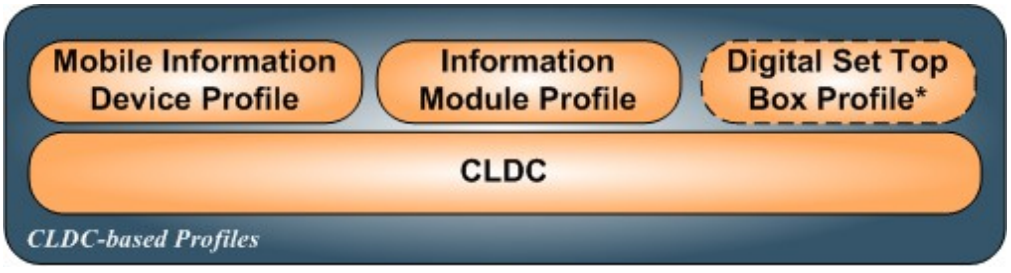
  - Όλα τα κινητής υποδιαστολής byte codes υποστηρίζονται από το CLDC 1.1
  - Προστέθηκαν οι classes Float και Double
  - Διάφορρες μέθοδοι έχουν προστεθεί στις βιβλιοθήκες για να των παραπάνω τάξεων, για να υποστηρίζουν τον χειρισμό τέτοιων τιμών
- Υποστήριξη ασθενών αναφορών (μικρά υποσύνολα των ασθενών αναφορών των κλάσεων της J2SE)
- Επανασχεδιασμένες, πιο συμβατές με την J2SE Calendar/Date/TimeZone κλάσεις
- Έχουν αποσαφηνιστεί οι απαιτήσεις χειρισμού σφαλμάτων κι έχει προστεθεί μία καινούρια class η NoClassDefFoundError
- Διάφορες μικρές αλλαγές στις βιβλιοθήκες κι επιδιόρθωση σε bug, όπως η πρόσθεση των παρακάτω μεθόδων και πεδίων
  - Boolean.TRUE και Boolean.FALSE
  - String.intern()
  - Date.toString()
  - Random.nextInt(int n)
- Η ελάχιστη απαιτούμενη μνήμη αυξήθηκε από 160 σε 192 kilobytes, κυρίως επειδή προστέθηκε η υποστήριξη των αριθμών κινητής υποδιαστολής
- Αυστηρότερη προδιαγραφή κειμένου και τα παρωχημένα σύνολα αφαιρέθηκαν
- Πολύ πιο αναλυτικός ελεγκτής προδιαγραφών

[15]

4.3.1.3 Technology Compatibility Kits

Τα CLDC TCKs είναι σουίτες τεστ, εργαλείων και τεκμηρίωσης που αποφασίζουν αν μία υλοποίηση προϊόντος συμβαδίζει με την CLDC προδιαγραφή [31].

Πάνω σε κάθε configuration βασίζονται κάποια profiles τα οποία δίνουν την δυνατότητα για την προσθήκη περισσότερων λειτουργιών στις φορητές συσκευές. Στην παρακάτω εικόνα, βλέπουμε ποια profiles βασίζονται στο CLDC.



Εικόνα 4.3 - Profiles τα οποία βασίζονται στο CLDC

[21]

4.3.2.1 CDC

Η Connected Device Configuration (CDC) τεχνολογία, η οποία έχει καθοριστεί από την JCP μέσω του JSR 36 και JSR 218 είναι ένα framework βασισμένο σε πρότυπα, για την ανάπτυξη εφαρμογών οι οποίες μπορούν να διαμοιραστούν μέσω ενός δικτύου συσκευών.

Στόχος του CDC είναι η υποστήριξη των χαρακτηριστικών από ένα ευρύ φάσμα συνδεδεμένων σε δίκτυο συσκευών, ενώ ταυτόχρονα, θα πρέπει να είναι εντός των περιορισμένων πόρων τους [8].

Το CDC καλύπετει τις συσκευές οι οποίες έχουν λιγότερο από 512 KB μνήμης για το JVM περιβάλλον. Η JVM η οποία χρησιμοποιείται ονομάζεται CVM. Ο garbage collector είναι ένα ξεχωριστό αντικείμενο το οποίο αναπτύσσεται από την CVM στις συσκευές. Τα threads είναι πλήρως υλοποιημένα μέσα στην CVM και είναι γνωστά ως Green Threads. Το class verification λαμβάνει χώρα μέσα στην CVM όπως ακριβώς και στην J2SE. Τα JSRs για το CDC είναι τα εξής:

- **CDC 1.0:** Το Connected Decive Configuration 1.0 (ή αλλιώς JSR 36) στοχεύει σε συσκευές οι οποίες έχουν μνήμη πάνω από 512 KB ROM και 128 KB RAM, συνδέονται σε δίκτυο, υποστηρίζουν πλήρη εγκατάσταση μίας JVM και έχουν user interface με περισσότερα χαρακτηριστικά από ότι το CLDC

- **CDC 1.1:** Το Connected Decive Configuration 1.1 (ή αλλιώς JSR 218) επεκτείνει το CDC 1.0 και απαιτεί μνήμη 256 KB RAM

[39]

Το CDC είναι ένα υπερσύνολο του CLDC και απευθύνεται σε συκευές με περισσότερες δυνατότητες από αυτό.

Η CDC προδιαγραφή ορίζει τέσσερα πράγματα:

- Τις δυνατότητες της JVM. Σε αντίθεση με το CLDC, η CDC VM είναι ένα πλήρης VM.

- Ένα υποσύνολο, από classes της έκδοσης J2SE 1.3

- Τα APIs για το Generice Connection Framework (GFC)

- Υποστήριξη για I/O, χρησιμοποιώντας τόσο το GCF καθώς και τις classes java.io και java.net πακέτα

Να σημειωθεί ότι το CDC δεν απαιτεί preverification των classes, καθώς ένα πλήρες class verification γίνεται στην συσκευή από την VM. Επειδή η VM που χρησιμοποιείται είναι με πλήρη χαρακτηριστικά, η ελάχιστη απαιτούμενη μνήμη για το CDC είναι 512 KB, καθώς και άλλα 256 KB για να μπορεί να

αποθηκεύει και να τρέχει εφαρμογές [8].

Πίνακας 4.2 - Τα Java Packages στο CDC

Όνομα	Περιγραφή
`java.lang`	Υποσύνολο της Java
`java.lang.ref`	Reference-object classes, which support a limited degree of interaction with the garbage collector
`java.lang.reflect`	Classes και interfaces για την απόκτηση reflective information πάνω στις classes και τα αντικείμενα
`java.math`	Υποσύνολο από classes για την τέλεση αριθμητικών πράξεων
`java.text`	Υποσύνολο από classes και interfaces για τον χειρισμό κειμένου, ημερομηνιών και μηνυμάτων, με τρόπο ανεξάρτητο από την χρησιμοποιούμενη γλώσσα
`java.io`	Υποσύνολο του system input and output μέσω data streams, serialization, και το file system
`javax.microedition.io`	Υποστήριξη δικτύου βασισμένη στο Generic Connection Framework, υποστήριξη για `file:` και `datagram:`
`java.util`	Υποσύνολο από collections, λειτουργίες ημερομηνίας και ώρας, internationalization, και διάφορες άλλες χρήσιμες classes
`java.util.zip`	Υποσύνολο από classes για την ανάγνωση αρχείων μορφής ZIP
`java.util.jar`	Υποσύνολο από classes για την ανάγνωση αρχείων της μορφής JAR
`java.net`	Υποσύνολο από classes για την υλοποίηση μίας δικτυακής εφαρμογής, υποστήριξη για datagram: και JAR I/O
`java.security`	Υποσύνολο από classes και interfaces framework ασφαλείας
`java.security.cert`	Υποσύνολο από classes και interfaces για parsing και διαχείριση πιστοποιητικών

[21]

Το CDC υποστηρίζει ορισμένα Profiles:

Εικόνα 4.4 - Profiles τα οποία βασίζονται στο CDC

- **Foundation Profile 1.1 (JSR 219)**
  - Core Java class library
  - Καμία υποστήριξη GUI
  - CLDC 1.1 compatibility library

- **Personal Basis Profile 1.1 (JSR 217)**
  - Υποστήριξη ελαφρών Components
  - Υποστήριξη xlet
  - Foundation Profile 1.1 APIs

- **Personal Profile 1.1 (JSR 216)**
  - Πλήρης υποστήριξη AWT
  - Υποστήριξη Applets
  - Migration path για Personal Java™ τεχνολογία
  - Personal Basis Profile 1.1 APIs

Επίσης, το CDC υποστηρίζει και ορισμένα Optional Packages:

- RMI Optional Package (JSR 66)
- JDBC Optional Package (JSR 169)
- Advanced Graphics and User Interface Optional Package (JSR 209)
- Java Secure Socket Extension (JSSE – JSR 219)
- Java Cryptography Extension (JCE – JSR 219)
- Java Authentication and Authorization Service (JAAS – JSR 219)

Application Models

Το CDC υποστηρίζει διαφορετικά application models ώστε να δώσει στους προγραμματιστές την ευελιξία να διαχειριστιούν όλες τις πιθανές ανάγκες των χρηστών.

- **Standalone** εφαρμογές οι οποίες μπορούν να υποστηρίξουν σχεδιασμούς οι οποίοι να διαχειρίζονται από μόνοι τους τον κύκλο ζωής και τους πόρους τους οποίους χρειάζονται

- **Managed εφαρμογές** όπως τα applets και τα xlets προσθέτουν ένα application management layer το οποίο αναλαμβάνει τις διεργασίες ανάπτυξης και διαχείρισης των πόρων

[36]

4.4 Profiles

4.4.1 Foundation Profile

Το Foundation Profile είναι αυτό το οποίο μπορεί να ονομαστεί "vertical" profile προδιαραφής. Ένα vertical profile διευκρινίζει τηη οικογένεια προϊόντων από το μέγεθος της μνήμης και τα προαιρετικά χαρακτηριστικά, τα οποία μπορούν να ενσωματώσουν περισσότερες λειτουργίες, εκτός από αυτό το profile. Τα GUI APIs είναι συχνά profiles τα οποία βασίζονται πάνω σε αυτό το profile. Τα JSRs για αυτό το profile είναι:

- **Foundation Profile 1.0:** Το Foundation Profile 1.0 (ή αλλιώς JSR 46) στοχεύει σε συσκευές με ελάχιστη μνήμη ROM 128 KB κι ελάχιστη RAM 512 KB, σύνδεση σε κάποιο δίκτυο, ενώ δεν παρέχεται κάποιο user interface

- **Foundation Profile 1.1:** Το Foundation Profile 1.1 (ή αλλιώς JSR 219) βασίζεται στο CDC 1.1 και υποστηρίζει J2SE 1.4 APIs. Οι συσκευές στις οποίες στοχεύει έχουν ελάχιστη μνήμη ROM 256 KB κι ελάχιστη RAM 512 KB. Ακόμα υποστηρίζει λειτουργικότητα για το Ipv6, ασφάλεια και μαθηματικές πράξεις

4.4.2 Personal Profile

Το Personal Profile παρέχει υψηλής πιστότητας σε web συσκευές οι οποίες υποστηρίζουν μία πλούσια σύνδεση δικτύου και μπορούν να χρησιμοποιήσουν applets Αν και προσωπικά προφίλ, JSR62, υποστηρίζει PersonalJava1.1.x και PersonalJava1.2.x του PersonalJava Εφαρμογή Περιβάλλον? Προφίλ Προσωπικά 1,1 υποστηρίζει την υιοθέτηση του υποσύνολο J2SE1.4 API ως προσωπικό προφίλ 1,0 βασίστηκε στο υποσύνολο των J2SE1. 3 API. Τα JSRs για αυτό το profile είναι:

- **Personal Profile 1.0:** Το Personal Profile 1.0 (ή αλλιώς JSR 62) στοχεύει σε συσκευές με 1,5 MB ελάχιστης μνήμης ROM και 1 MB ελάχιστης μνήμης RAM, μια ισχυρή σύνδεση δικτύου η οποία βασίζεται στα CDC 1.0 και Foundation Profile 1.0 ενώ επιτρέπει στις εφαρμογές οι οποίες έχουν αναπτυχθείμε χρήση pJava Emulation Environment Specification 1.1.x και 1.1 .2 να τρέχουν σε συσκευές οι οποίες χρησιμοποιούν αυτό το profile

- **Personal Profile 1.1:** Το Personal Profile 1.0 (ή αλλιώς JSR 216) επεκτείνει το Personal Profile 1.0 και αναβαθμίζει στα APIs σε J2SE 1.4 καθιστώντας βέβαιη την χρήση του Graphics πακέτου. Οι στοχευόμενες συσκευές πρέπει να έχουν ελάχιστη μνήμη ROM και RAM στα 3,5 MB

4.4.3 Personal Basis Profile

Αυτό το profile παρέχει ένα βασικό user interface και βασίζεται στο CDC και στο Personal Profile. Επίσης παρέχει τις xlet classes οι οποίες απαιτούνται για την JavaTV. Τα JSRs για αυτό το profile είναι:

- **Personal Basis Profile 1.0:** Το Personal Basis Profile 1.0 (ή αλλιώς JSR 129) στοχεύει σε συσκευές με ελάχιστη μνήμη ROM 2MB κι ελάχιστη RAM 1 MB. Τα AWT APIs του βασίζονται στο JDK 1.1 ώστε να είναι πιο ελαφρύ

- **Personal Basis Profile 1.1:** Το Personal Basis Profile 1.1 (ή αλλιώς JSR 217) επεκτείνει το Personal Basis Profile 1.0 και αναβαθμίζει όλα τα APIs σε J2SE 1.4 για να είναι συμβατά με τα προαιρετικά APIs γραφικών, όπως το AGUI

4.4.4 Information Module Profile

Το Information Module Profile υποστηρίζει συσκευές συνδεδεμένες συσκευές στο δίκτυο χωρίς την χρήση user interface. Στοχεύει σε συσκευές, όπως modems, οικιακές συσκευές, καθώς και βιομηχανικές συσκευές μέτρησης. Τα JSRs για αυτό το profile είναι:

- **Information Module Profile:** Το Information Module Profile (ή αλλιώς JSR 195) είναι ένα υποσύνολο του MIDP 1.0, με την εξαίρεση πως του λείπουν οι LCDUI classes.

- **Information Module Profile - Next Generation:** Το Information Module Profile Next Generation (ή αλλιώς JSR 228) είναι ένα υποσύνολο του MIDP 2.0, με την εξαίρεση πως του λείπουν οι LCDUI classes.

4.4.5 Digital Set Top Box Profile

Αυτό το profile είναι ένα υποσύνολο της JavaTV και στοχεύει για τα μικρά set-top boxes τα οποία ανατπύσσουν το OCAP. Αυτό το profile διαφέρει από την JavaTV λόγω του γεγονότος ότι βασίζεται στο CLDC, ώστε να υποστηρίξει συσκευές με μικρότερη μνήμη. Το JSR για αυτό το profile είναι:

- **Digital Set Top Box Profile:** Το Digital Set Top Box Profile (ή αλλιώς JSR 242) είναι ένα υποσύνολο τους Personal Java Profile και της JavaTV και βασίζεται στις classes του OCAP. Τα xlets τα οποία έχουν αναπτυχθεί με αυτό το profile είναι προς τα πάνω συμβατά με την JavaTV

..

4.4.6 Mobile Information Device Profile

Το Mobile Information Device Profile επεκτείνει κι ενισχύει το CLDC για να παράσχει ένα Java περιβάλλον για την ανάπτυξη εφαρμογών για κινητές συσκευές. Τα JSRs για αυτό το προφίλ είναι:

- **MIDP 1.0:** Το Mobile Information Device Profile 1.0 (ή αλλιώς JSR 37) επεκτείνει το CLDC1.0 για να υποστηρίξει τα MIDlets παρέχοντας ένα GUI toolkit, αποθήκευση δεδομένων, αποστολή μηνυμάτων, στοιχεία δικτύωσης και ασφάλεια.

- **MIDP 2.0:** Το Mobile Information Device Profile 2.0 (ή αλλιώς JSR 118) επεκτείνει το MIDP 1.0 ώστε να περιλαμβάνει https, domain μοντέλο ασφαλείας με ψηφιακά πιστοποιητικά, ενώ ακόμη περιλαμβάνει OTA provisioning, push αρχιτεκτονική στα MIDlets ούτως ώστε να αναποκρίνονται στα events, μικρός xml parser και ένα βασικό API ήχουν, ενώ προστείθεταιhorizontal και η λειτουργικότα παιχνιδιών με το LCDUI

- **MIDP 3.0:** Το Mobile Information Device Profile 3.0 (ή αλλιώς JSR 271) επεκτείνει το MIDP2.0 ώστε να επιτρέψει στα MIDlets να ξεκινούν μόνα τους, να τρέχουν στο background μέσα στην ίδια VM, και να επιτρέπεται η επικοινωνία μεταξύ τους. Η εκτεταμένη λειτουργικότητα περιλαμβάνει ασφαλή αποθήκευση και διαγραφή εγγραφών, την χρήση του IPv6, πολλαπλές διασυνδέσεις δικτύου ανά συσκευή

[39]

4.5 Optional Packages

4.5.1 Η J2ME Platform

Όταν η Java 2 Platform, Micro Edition (J2ME) πρωτοεμφανίστηκε, μόνο ένα configuration το Connected Limited Device Configuration (CLDC), και ένα profile, the Mobile Information Device Profile (MIDP) είχαν οριστεί σαν επίσης προδιαγραφές (specifications) στην Java Community Process (JCP). Πλέον υπάρχουν περίπου 300 προδιαγραφές σε διάφορα στάδια στην JCP, και πολλές από αυτές ορίζουν προαιρετικά πακέτα, αντί για configurations ή profiles.

Αρχικά η J2ME platform αποτελούνταν μόνο από configurations and profiles. Ο σκοπός των προδιαγραφών της J2ME επικεντρώθηκε στην ανάπτυξη νέων profiles. Καθώς η πλατφόρμα εξελισόταν, έγινε εμφανές στους προγραμματιστές ότι χρειαζόταν και μία 3η κατηγορία του J2ME component, το προαιρετικό πακέτο. Ήταν επίσης ξεκάθαρο ότι η συνολική αρχιτεκτονική της J2ME πλατφόρμας χρειαζόταν πιο ακριβή ορισμό. Η προδιαγραφ JSR 68 ορίζει J2ME configurations, profiles και προαιρετικά πακέτα σε όρους χαμηλού επιπέδου στοιχείων.

Το JSR 68 είναι πρωταρχικού ενδιαφέροντος για του ειδικούς, ορίζοντας άλλες J2ME προδιαγραφές. Ένας από τους σκοπούς του, είναι να αποτρέψει την δημιουργία παρόμοιων, αλλά μη συμβατών APIs. Η επαναχρησιμοποίηση των υπαρχόντων APIs, είτε αυτά ορίζονται από την J2ME, J2SE ή J2EE, ενθαρρύνεται όποτε είναι δυνατό. Η επαναχρησιμοποίηση προλαμβάνει τον μη απαραίτητο κατακερματισμό της Java platform σαν σύνολο και προωθεί την μεταφερσιμότητα του κώδικα στις τρεις εκδόσεις.

4.5.2 Χρησιμοποιώντας τα Optional Packages

Για την χρήση των προαιρετικών πακέτων θα πρέπει να γίνει έλεγχος για την υποστήριξή τους στην συσκευή. Αν δεν υποστηρίζεται τότε θα πρέπει να ληφθεί αυτό υπόψην από τον προγραμματιστή για να γίνει ομαλή έξοδος από το πρόγραμμα. Αφήνοντας να προκληθεί εξαίρεση, δημιουργείται σύγχυση στον χρήστη για τον πρόγραμμα.

```
...
public static boolean isWMAPresent(){
    try {
        Class.forName(
               "javax.wireless.messaging.MessageConnection" );
        return true;
    }
    catch( Exception e ){
        return false;
    }
}
…
```

[6]

Παρακάτω είναι η λίστα με τα optional packages

- Information Module Profile (IMP), JSR 195
- Wireless Messaging API (WMA); JSR 120, JSR 205
- Mobile Media API (MMAPI), JSR 135
- Location API for J2ME, JSR 179
- SIP API for J2ME, JSR 180

- Security and Trust Services API for J2ME, JSR 177
- Mobile 3D Graphics, JSR 184
- J2ME Web Services APIs (WSA), JSR 172
- Bluetooth API, JSR 82
- J2ME RMI, JSR 66
- JDBC for CDC/Foundation Profile API, JSR 169

[32]

Με την φράση optional package (προαιρετικό πακέτο) εννοούμε πως ο κατασκευαστής μίας συσκευής είναι εκείνο ο οποίος πρέπει να υποστηρίξει το API στην συσκευής. Σε περίπτωση που δεν το κάνει, τότε δεν μπορεί ούτε κάποιος χρήστης, ούτε κάποιος προγραμματιστής να εγκαταστήσει από μόνος του το package αυτό. Έτσι δεν θα λειτουργήσει μία εφαρμογή η οποία το χρησιμοποιεί [5].

4.6 Ανακεφαλαίωση

Σε αυτό το κεφάλαιο είδαμε τι είναι τα Configurations, τα Profiles και τα Optional Packages. Πλέον ο χρήστης έχει σε μεγάλο βαθμό επίγνωση της δομής της J2ME. Στο κεφάλαιο που ακολουθεί, ο χρήστης θα μάθει για το Mobile Information Device Profile (MIDP) και τα MIDlets, ώστε να μπορεί να δημιουργεί εφαρμογές.

Κεφάλαιο 5ο

5.1 Εισαγωγή

Σε αυτό το κεφάλαιο θα δούμε τα πιο γνωστό Profile της J2ME, το Mobile Information Device Profile (MIDP), με το οποίο φτιάχνονται εφαρμογές για τις κινητές συσκευές. Ακόμα ο αναγνώστης θα πληροφορηθεί πως λειτουργεί ο κύκλος ζωής του MIDlet, της κύριας δηλαδή class μίας εφαρμοφής. Επίσης, θα δούμε και τα διάφορα επίπεδα γραφικού περιβάλλοντος και πως οργανώνονται αυτά.

5.2 Mobile Information Device Profile

5.2.1 Εισαγωγή στο MIDP

Το Mobile Information Device Profile (MIDP) για την Java ™ 2 Platform, Micro Edition (J2ME ™) είναι μια αρχιτεκτονική κι ένα σύνολο από Java βιβλιοθήκες οι οποίες δημιουργούν ένα ανοικτό περιβάλλον εφαρμογής για τις μικρές, με περιορισμένους πόρους κινητές συσκευές πληροφορίών (mobile information devices, ή MIDs) [23]. Το MIDP είναι στην κορυφή του Connected Limited Device Configuration (CLDC) και αποτελεί ένα σύνολο από χαμηλού επιπέδου διασυνδέσεις προγραμματισμού. Η προδιαγραφή αυτή, αναπτύχθηκε υπό την Java Community Process σαν JSR 37 (MIDP 1.0) και JSR 118 (MIDP 2.0). Από το 2007, το MIDP 3.0 αναπτύσεται υπό το JSR 271 [51].

Τυπικά παραδείγματα MIDP συκευών είναι τα κινητά τηλέφωνα, οι pagers, και τα ασύρματα organizers. Οι ελάχιστες απαιτήσεις που πρέπει να έχει μία συσκευή για να υποστηρίξει το MIDP είναι:

Το MIDP μαζί με το CLDC παρέχει ένα ολοκληρωμένο περιβάλλον εκτέλεσης J2ME εφαρμογών, οι οποίες στοχεύουν σε κινητές συσκευές. Τι περιβάλλον επιτρέπει νέες εφαρμογές και υπηρεσίες να αναπτύσσονται στις συσκευές των τελικών χρηστών. Με άλλα λόγια, το MIDP αφήνει τους προγραμματιστές να επικεντρωθούν στην λειτουργικότητα των εφαρμογών, ενώ το MIDP φροντίζει για την λειτουργικότητα στις διαφορετικές πλατφόρμες κινητής. Ο προσδιορισμός του MIDP απευθύνεται σε θέματα όπως οι διασυνδέσεις χρηστών, η αποθήκευση δεδομένων, το δίκτυο και το μοντέλο εφαρμογής [23].

Οι απαιτήσεις για να υποστηρίξει μία συσκευή το MIDP

Μνήμη:

- 256 KB ROM για το MIDP
- 8 KB ROM για τα δεδομένα της εφαρμογής

- 128 KB RAM για την VM

Display:

- Μέγεθος οθόνης: 96x45
- Βάθος display: 1-bit
- Σχήμα pixel: περίπου 1:1

Input

- Ένας από τους παρακάτω μηχανισμούς
- Πληκτρολόγιο για ένα χέρι
- Πληκτρολόγιο για δύο χέρια
- Οθόνη αφής

Δίκτυο:

- Ασύρματο, με περιορισμένο bandwidth

Ήχος:

- Ικανότητα να παίζει ήχους είτε μέσω υλικού είτε μέσω λογισμικού

5.2.2 Περιοχές τις οποίες καλύπτει η προδιαγραφή MIDP

Η MIDP προδιαγραφή επεκτείνει τη λειτουργικότητα η οποία ορίζεται από την CLDC προδιαγραφή. Η MIDP προδιαγραφή ορίζει ένα σύνολο από APIs τα οποία προσθέτουν ένα ελάχιστο σύνολο από δυνατότητες οι οποίες είναι κοινές σε διάφορα είδη κινητών συσκευών. Οι τομείς οι οποίοι καλύπτονται από το MIDP είναι:

- μοντέλο εφαρμογής (το οποίο ορίζει πως θα ελέγχεται μία εφαρμογή)
- υποστήριξη user interface (LCDUI)
- υποστήριξη δικτύου
- αποθήκευση δεδομένων
- ήχοι
- 2D παιχνίδια
- ασφάλεια μέσω HTTPS και secure sockets
- διάφορες classes όπως χρονμετρητές και exceptions

Εκτός από τους παραπάνω τομείς, η MIDP προδιαγραφή καθορίζει την επέκταση του μοντέλου

εφαρμογής του CLDC η οποία επιτρέπει την εκτέλεση και την επικοινωνία των εφαρμογών που ονομάζεται MIDlets. Ένα MIDlet είναι η βασική μονάδα εκτέλεσης στο MIDP.

5.2.3 Πλεονεκτήματα του MIDP

Το mobile user interface

Το MIDP έχει στα χαρακτηριστικά του ένα interface API το οποίο προφυλάσσει τους προγραμματιστές από την πολυπλοκότητα της κατασκευής φορητών εφαρμογών. Αυτό το υψηλού επιπέδου API επιτρέπει στους προγραμματιστές να δημιουργήσουν εύκολες προς χρήση, με εξαιρετικά γραφικά, εφαρμογές για φορητές συσκευές μειώνοντας κιόλας τον χρόνο ανάπτυξης τους. Η λειτουργικότητα του user interface περιλαμβάνει predefined οθόνες, επιλογές από μία λίστα, pop up alert dialogs, κινούμενα tickers και τροποποποίηση κειμένου. Τα forms είναι μία μορφή οθόνης με πολλά predefined αντικείμενα, όπως textfields, date fields, διαγράμματα, καθώς κι αντικείμενα τα οποία μπορεί να έχει φτιάξει ο προγραμματιστής. Όλες οι οθόνες αναγνωρίζουν την συσκευή στην οποία βρίσκονται για την υποστήριξη της ανάλυσης οθόνης, για την πλοήγηση μέσα στην συσκευή.

Λειτουργικότητα Multimedia και παιχνιδιών

Το MIDP είναι ιδανικό για την δημιουργία φορητών πανχνιδιών και multimedia εφαρμογών. Ένα χαμηλού επιπέδου, user interface συμπληρώνει το υψηλού επιπέδου API, δίνοντας στους προγραμματιστές δίνοντας μεγαλύτερο έλεγχο των γραφικών. Το Game API προσθέτει συγκεκριμένες λειτουργίες στα παιχνίδια, όπως τα sprites και τα tiled layers. Ακόμα μπορεί να προστεθεί ήχος και αρχεία WAV μορφής. Ακόμα με την χρήση του Mobile Media API (MMAPI), μπορούν να προστεθεί multimedia περιεχόμενο.

Εκτεταμένη συνδεσιμότητα

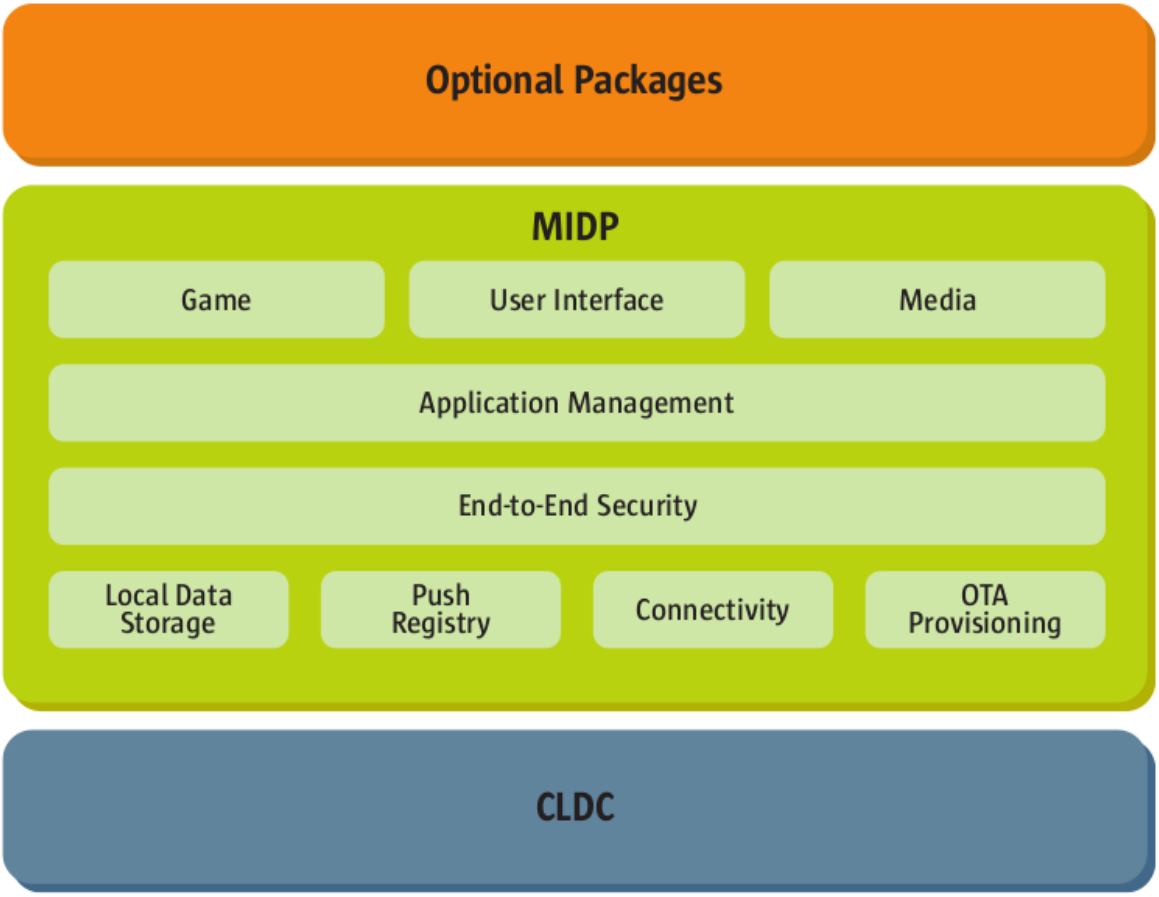
Το MIDP υποστηρίζει πρωτοπόρα πρότυπα σύνδεσης, συμπεριλαμβανομένων και των HTTP, HTTPS, datagram, sockets, server sockets και serial port επικοινωνία. Το MIDP επίσης υποστηρίζει SMS και CBS ικανότητες σε GSM και CDMA δίκτυα, μέσω του Wireless Messaging API (WMA). Το MIDP υποστηρίζει, επίσης, ένα server push μοντέλο. Ένα push registry παρακολουθεί τις εφαρμογές οι οποίες έχουν γίνει registered ώστε να λαμβάνουν εισερχόμενες πληροφορίες από το δίκτυο. Όταν φτάσει η πληροφορία, η συσκευή αποφασίζει για το αν θα ξεκινήσει την εφαρμογή με βάση τις προτιμήσεις του χρήστη. Αυτή η push αρχιτεκτονική επιτρέπει στους προγραμματιστές να περιλαμβάνουν alets, μηνύματα, και broadcasts στις MIDP εφαρμογές,

Over-the-air provisioning

Ένα σημαντικό όφελος του MIDP είναι η δυνατότητα δυναμικής ανάπτυξης και της ανανέωσης εφαρμογών over the air (OTA). Η προδιαγραφή MIDP ορίζει τον τρόπο με τον οποίο οι εφαρμογές ανακαλύπτονται, εγκαθίστανται, ανανεώνονται, και αφαιρούνται στις κινητές συσκευές. Το MIDP επίσης επιτρέπει την αναγνώριση για το ποιες εφαρμογές δουλεύουν σε μία συγκεκριμένη συσκευή. Το MIDP OTA Provisioning μοντέλο, έχει αναπτυχθεί και υιοθετηθεί από τους μεγαλύτερου κατασκευαστές, ώστε ν υπάρχει μία αξιόπιστη και ασφαλής λύση.

Από άκρη σε άκρη ασφάλεια

Για την προστασία του δικτύου, το MIDP παρέχει ένα ισχυρό μοντέλο ασφαλείας βασισμένο σε ανοιχτά πρότυπα. Η χρήση του HTTPS βασίζεται σε υπάρχοντα πρότυπα όπως το Secure Sockets Layer (SSL) και Wireless Transport Layer Security (WTLS) για να επιτραπεί η μεταφορά των κρυπτογραφημένων δεδομένων. Τα security domain παρέχουν προστασία ενάντια στην μη εξουσιοδοτημένη πρόσβαση των δεδομένων, των εφαρμογών, και άλλων δικτύων και πόρων στην συσκευή. Από προεπιλογή, οι MIDP εφαρμογές δεν είναι αξιόπιστες, και καταχωρούνται στα μη έμπιστα domains ώστε να εμποδίζουν την πρόσβαση στους πόρους της συσκευής. Για να αποκτήσει πρόσβαση, μια εφαρμογή MIDP πρέπει να ανατεθεί σε ένα συγκεκριμένο domain το οποίο να ορίζεται στην κινητή συσκευή, και είναι σωστά υπογεγραμμένο χρησιμοποιούντα το πρότυπο ασφαλείας X.509 PKI [33].



Εικόνα 5.1 - MIDP υποσυστήματα και υπηρεσίες

5.2.4 Το user interface στο MIDP

Υπό το πρίσμα των μεγάλων διακυμάνσεων στα κινητά τηλέφωνα και σε άλλες συσκευές στις οποίες απευθύνεται το MIDP, οι απαιτήσεις για το user interface είναι πραγματικά μία πρόκληση. Οι MIDP συσκευές διαφέρουν από τα desktop συστήματα με πολλούς τρόπους, κυρίως στον τρόπο με τον οποίο ο χρήστης αλληλεπιδρά με αυτά. Παρακάτω παρατείθενται οι λόγοι για τους οποίους το MIDP σχεδιάστηκε με αυτόν τον τρόπο:

- Οι συσκευές και οι εφαρμογές θα πρέπει να είναι χρήσιμες στους χρήστες, οι οποίοι δεν είναι απαραίτητα ειδικοί στη χρήση υπολογιστών

- Οι συσκευές και οι εφαρμογές θα πρέπει να είναι χρήσιμες σε περιπτώσεις κατά τις οποίες ο χρήστης δεν μπορεί να δώσει μεγάλη προσοχή. Για παράδειγμα, πολλά κινητά τηλέφωνα και άλλες ασύρματες συσκευές χειρίζονται με το ένα χέρι, ενώ ο χρήστης περπατάει, μαγειρεύει ή κάνει οτιδήποτε άλλο

- Οι μονάδες I/O είναι πολύ διαφορετικές σε σχέση με τους υπολογιστές

Λόγω των παραπάνω το MIDP παρέχει το `javax.microedition.lcdui` πακέτο για την δημιουργία του user interface.

[56]

5.3 Limited Capability Device User Interface

5.3.1 Εισαγωγή στο Limited Capability Device User Interface

Το Limited Capability Device User Interface (LCDUI) έχει μια απλή οθόνη ως προσέγγιση προς το user interface του χρήσητη. Ένα και μοναδικό Displayable είναι ενεργό κατά την διάρκεια εκτέλεσης της εφαρμογής. Το LCDUI API παρέχει ένα μικρό σύνολο από κοινά displayables όπως: List, Alert, TextBox, Form και Canvas. Για όλα τα displayables η υλοποίηση του MIDP η οποία υπάρχει στην συσκευή έχει τον έλεγχο κατά την παρουσίαση και την εμφάνιση του displayable. Η Canvas είναι μία χαμηλού επιπέδου επιφάνεια γραφικών για την οποία, μία εφαρμογή έχει τον πλήρη έλεγχο σε αυτό το οποίο φαίνεται αν και κανονικά, φυλάσσεται και κάποιος χώρος στην οθόνη για τον τίτλο και τους δείκτες με τους οποίους χρησιμοποιείται μια εφαρμογή.

Το LCDUI διαθέτει επίσης αρκετά μοναδική προσέγγιση στις abstract λειτουργίες, οι οποίες ονομάζεται Commands. Η τοποθέτηση των εντολών οι οποίες προστίθενται σε ένα displayable βασίζεται εντελώς στο στην υλοποίηση αυτού του toolkit. Η ιδέα του abstraction των εντολών, είναι να κάνει τις εφαρμογές πιο φορητές μεταξύ διαφορετικών κινητών συσκευών [49].

5.3.2 Classes, Interfaces και χαρακτηριστικά

Πίνακας 5.1 - Components του LCDUI

Class	*Περιγραφή*
Interfaces	
Choice	Παρέχει το κοινό interface για την διαχείριση της επιλογής των αντικειμένων
CommandListener	Δημιουργεί ένα listener για command events από το high-level UI
ItemStateListener	Δημιουργεί ένα listener για αλλαγές στην κατάσταση ενός αντικειμένου Item
UI System & Utility Classes	
Display	Αντιπροσωπεύει το display manager και τις input συσκευές του συστήματος
Font	Φέρνει τα font αντικείμενα
Image	Παρεχει μία class για να κρατάει τα δεδομένα εικόνας (σε PNG μορφή)
AlertType	Παρέχει μία class βοηθό, η οποία ορίζει τους τύπους των Alerts, όπως *ALARM, CONFIRMATION, ERROR, INFO, WARNING*
Displayable	Παρέχει μία βασική abstract class για ένα αντικείμενο το οποίο μπορεί να γίνει display
High-Level UI	
Command	Παίρνει τις εντολές του χρήστη στο UI
Screen Classes	
Screen	Παρέχει μία βασική class για high-level UI αντικείμενα
Alert	Παρέχει μία οθόνη για να ειδοποιήσει τον χρήστη για κάτι
List	Παρέχι ένα αντικείμενο οθόνης, το οποίο περιέχει μία λίστα από επιλογές
TextBox	Παρέχει ένα αντικείμενο οθόνης, το οποίο χρησιμοποιείται για τροποποποίηση κειμένου

Forms & Items	
Form	Παρέχει μία οθόνη η οποία λειτουργεί σαν container για ένα ή περισσότερα Items
Item	Παρέχει μία βασική class για αντικείμενα τα οποία μπορούν να μπουν σε μία Form ή Alert
ChoiceGroup	Παρέχει ένα UI αντικείμενο για την παρουσίαση μίας λίστας με επιλογές
DateField	Παρέχει ένα UI αντικείμενο στο οποίο ο χρήστης βάζει την ημερομηνία
Gauge	Παρουσιάζει μία μπάρα για να δείχνει την πρόοδο
ImageItem	Παρέχει ένα Item το οποίο είναι επίσης και εικόνα
StringItem	Παρέχει ένα Item αντικείμενο για την αναπαράσταση ενός String
TextField	Παρέχει ένα Item το οποίο χρησιμοποιείται για εισαγωγή κειμένου
Ticker	Παρέχει ένα Item το οποίο εμφανίζει στην οθόνη ένα κείμενο σε κίνηση
Low-Level UI	
Graphics	Παρέχει 2D εργαλεία γραφικών
Canvas	Παρέχει την βασική class για την δημιουργία low-level UI γραφικών

System ή utility classes

- Display
- Font
- AlertType
- Ticker

Low-level API classes

- Canvas
- Graphics

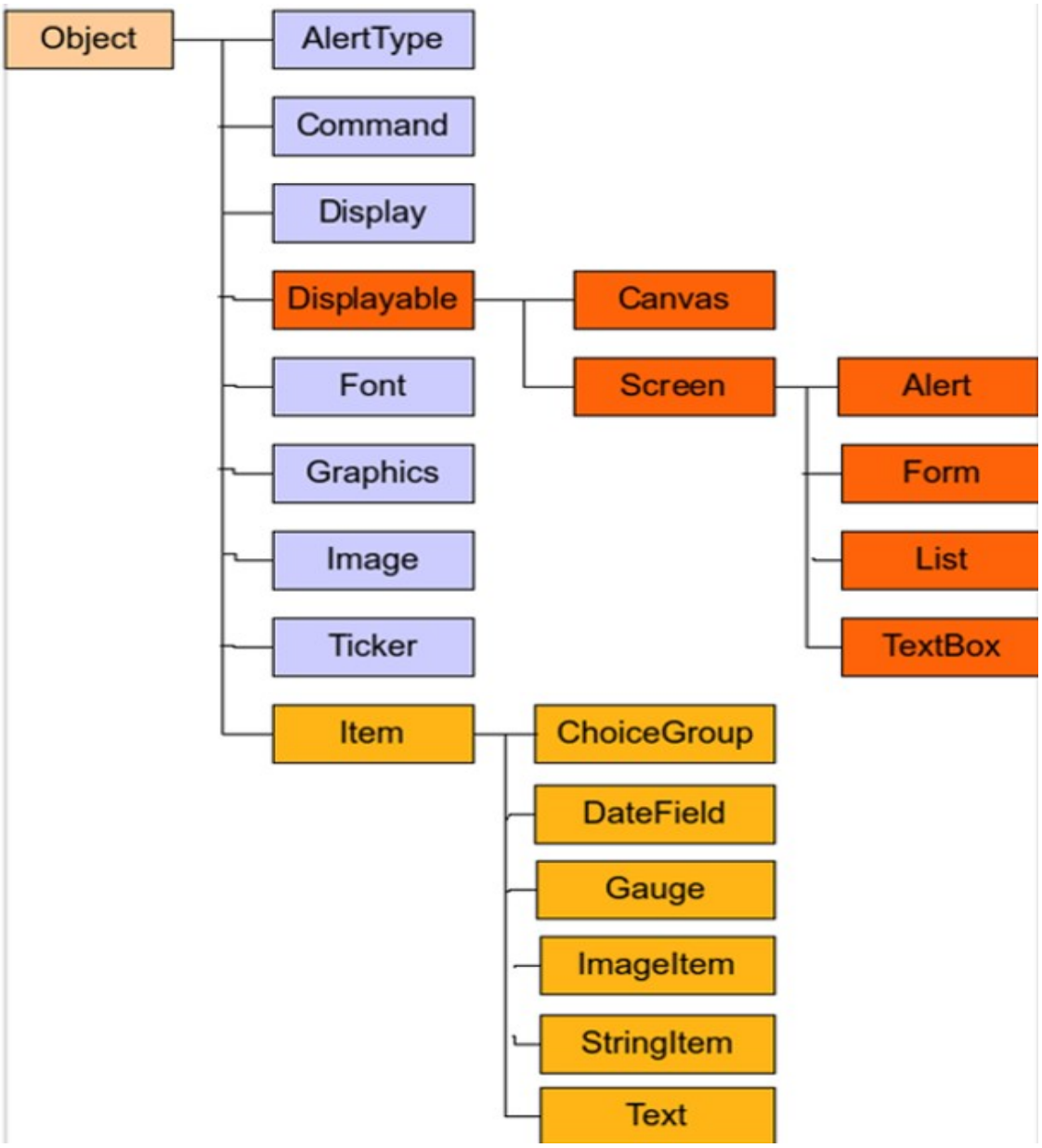
High-level API Screen classes

- Alerts
- Form
- List
- TextBox

High-level API Form component classes

- ChoiceGroup

- DateField
- Gauge
- ImageItem
- StringItem
- TextField



Εικόνα 5.2 - Η class ιεραρχία του LCDUI

[38]

5.4 MIDlet

5.4.1 Εισαγωγή στα MIDlets

Οι Java™ εφαρμογές οι οποίες τρέχουν σε MIDP συσκευές ονομάζονται MIDlets. Τα MIDlets είναι φορητά σε όλες τις συμβατές συσκευές οι οποίες υποστηρίζουν τα ίδια προαιρετικά APIs τα οποία χρησιμοποιούν τα MIDlets.

Για να θεωρηθεί σαν MIDlet μία εφαρμογή Java για κινητά, θα πρέπει:

- Να κληρονομεί την class javax.microedition.midlet.MIDlet η οποία ελέγχει τον κύκλο ζωής ενός MIDlet
- Να βρίσκεται σε ένα JAR αρχείο
- Να περιέχει ένα MANIFEST.MF αρχείο μέσα στο .jar αρχείο
- Να περιέχει ένα JAD αρχείο
- Να έχει κάνει preverify όλα τα .class αρχεία πριν την ανάπτυξη.

Τα MIDlets έχουν σχεδιαστεί για να υποστηρίζουν μια συγκεκριμένη έκδοση MIDP και είναι εγγυημένα στο να λειτουργήσουν σωστά με τις κινητές συσκευές οι οποίες υποστηρίζουν αυτήν την έκδοση. Τα MIDlets μπορούν επίσης να χρησιμοποιούν στοιχεία από προαιρετικά APIs, τα οποία μπορεί να μην υποστηρίζονταιαπό την κινητή συσκευή. Στις περιπτώσεις αυτές, η λειτουργικότητα των προαιρετικών APIs δεν υποστηρίζεται. Όταν ένα MIDlet πληροί τις παραπάνω απαιτήσεις των και είναι συμβατό με την έκδοση του MIDP και με τα προαιρετικά APIs τα οποία χρησιμοποιεί η φορητή συσκευή, μπορεί να εγκατασταθεί και να εκτελεστεί σε αυτή την κινητή συσκευή σωστά. Η εγκατάσταση, απεγκατάσταση και οι όποιες ενέργειες κατά το κατά τον χρόνο εκτέλεσης χειρίζονται από το Application Management Software (AMS).

5.4.2 MIDlet suites

Ένα MIDlet suite είναι μια συλλογή ενός ή περισσοτέρων MIDlets σε ένα ενιαίο αρχείο JAR ώστε να έχουν μεγαλύτερη δυνατότητα στο να μοιραστούν εξωτερικούς πόρους, παρά σαν ξεχωριστά MIDlets. Όλα τα MIDlets σε ένα MIDlet suite εγκαθείστανται στην συσκευή ως μια ενιαία οντότητα. Κατά το χρόνο εκτέλεσης, αν η συσκευή υποστηρίζει την ταυτόχρονη λειτουργία περισσοτέρων του ενός MIDlet, όλων τα active MIDlets από ένα MIDlet suite, εκτελούνται στην ίδια Java VM. Άρα, όλα τα MIDlets στο ίδιο suite διαμοιράζονται τα ίδια instances όλων των Java classes και των υπολοίπων πόρων οι οποίοι φορτώνονται στην Java VM. Αυτό πρακτικά σημαίνει ότι τα δεδομένα μπορούν να διαμοιράζονται μεταξύ MIDlets και ότι μπορεί να χρησιμοποιηθεί ο συγχρονισμός για την προστασία από την ταυτόχρονη πρόσβαση.

Ένα MIDlet δεν μπορεί να ξεκινήσει από ένα άλλο MIDlet και ένα MIDlet suite δεν μπορεί να ξεκινήσει ένα άλλο MIDlet suite. Επιπλέον, δεν μπορούν να διαμοιραστούν πόροι μεταξύ MIDlets τα οποία είναι σε διαφορετικά suites. Μόνο τα persistent δεδομένα τα οποία χρησιμοποιούν το Record Management System (RMS) μπορούν να έχουν ταυτόχρονη πρόσβαση από διαφορετικά MIDlet suites

[16].

5.4.3 MIDlet Life Cycle

Ο κύκλος ζωής ενός MIDlet καθορίζει τις καταστάσεις εκτέλεσης ενός MIDlet, καθώς και τις έγκυρες καταστάσεις μετάβασης. Ο κύκλος ζωής μπορεί να θεωρηθεί ότι ξεκινάει όταν το MIDlet είναι εγκατεστημένο σε μια συσκευή και διαρκεί μέχρι την απεγκατάσταση του. Οι περισσότερες λειτουργίες του κύκλου ζωής είναι συγκεντρωμένες την στιγμή κατά την οποία το MIDlet ξεκινάει, είναι σε κατάσταση active ή κλείνει. Όταν ένα MIDlet suite είναι εγκατεστημένο σε μια συσκευή, οι classes, κι ό,τι άλλο είναι απαραίτητο για την εκτέλεσή του, φυλάσσονται μέσα στην συσκευή και είναι έτοιμα προς χρήση. Τα MIDlets διαχειρίζονται μέσα από το Application Manager της συσκευής [17].

Παρακάτω περιγράφεται ο κύκλος ζωής ενός MIDlet με τις έγκυρες καταστάσεις του:

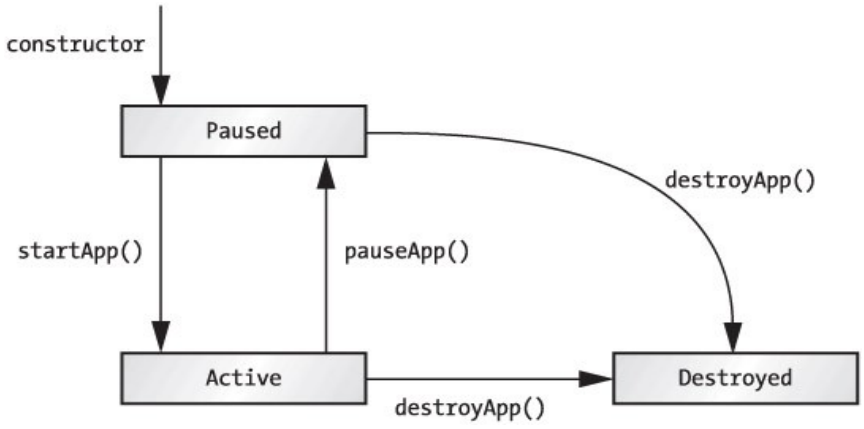
Όταν το MIDlet πρόκειται να εκτελεστεί, ένα instance του δημιουργείται. Όταν τρέχει ο constructor του MIDlet τότε αυτό μπαίνει στην κατάσταση Paused

Στη συνέχεια, το MIDlet εισέρχεται στην Active κατάσταση αφού ο application manager καλεί την μέθοδο startApp ()

Ενώ το MIDlet είναι Active, ο application manager μπορεί να αναστείλει την εκτέλεσή του, καλώντας την μέθοδο pauseApp (). Αυτό θέτει το MIDlet πίσω στην κατάσταση Paused. Ένα MIDlet μπορεί να θέσει τον εαυτό του στην κατάσταστη Paused καλώντας την μέθοδο notifyPaused()

Ενώ το MIDlet είναι σε κατάσταση Paused, ο application manager μπορεί να καλέσει την μέθοδο startApp () για να το θέσει και πάλι στην κατάσταση Active.

Ο application manager μπορεί να τερματίσει την εκτέλεση του MIDlet καλώντας την μέθοδο destroyApp (), στο οποίο σημείο το MIDlet καταστρέφεται και περιμένει τον garbage collector. Ένα MIDlet μπορεί να καταστρέψει τον ευατό του, καλώντας την μέθοδο notifyDestroyed().



Εικόνα 5.3 - Κύκλός ζωή τους MIDlet

Υπάρχει μία ακόμη μέθοδος στην MIDlet class και είναι σχετική με τη διάρκεια του κύκλου ζωής του MIDlet: η μέθοδος resumeRequest (). Ένα MIDlet σε Paused κατάσταση μπορεί να καλέσει αυτή τη μέθοδο για να δείξει στον application manager ότι θέλει να γίνει Active. Μπορεί να φαίνεται παράξενο να σκεφτούμε για ένα MIDlet ότι στην Paused κατάσταση μπορεί να τρέξει οποιοδήποτε κομμάτι κώδικα. Ωστόσο, τα Paused MIDlets είναι ακόμη σε θέση να χειρίζονται events ή άλλες μορφές callbacks και, συνεπώς, έχουν κάποιες πιθανότητες να καλέσουν την μέθοδο resumeRequest (). Εάν ο application manager αποφασίσει να κάνει ένα από τα Paused MIDlets Active το κάνει μέσω τού συνήθους μηχανισμού της καλώντας την μέθοδο startApp ().

5.5 Ανακεφαλαίωση

Σε αυτό το κεφάλαιο είδαμε το Profile MIDP, καθώς και την κύρια class μίας φαρμογής J2ME. Ακόμα ο ανανγνώστης είδε σε μεγάλο βαθμό τα επίπεδα του γραφικού περιβάλλοντος.

Κεφάλαιο 6ο

6.1 Εισαγωγή

Σε αυτό το κεφάλαιο θα παρουσιαστεί το JSR 75, το οποίο αποτελείται από δύο μέρη, το FileConnection Optional Package και το PIM Optional Package. Το πρώτο είναι απαραίτητο για την δημιουργία, εγγραφή και ανάγνωση αρχείων και το δεύτερο για την προσπέλαση, την ανάγνωση, εγγραφή και επεξεργασία των Contacts, Events και ToDos μίας κινητής συσκευής.

6.2 Το FileConnection Optional Package

6.2.1 Εισαγωγή στο FileConnection Optional Package

Οι νέοι προγραμματιστές στην J2ME συχνά εκπλήσσονται όταν ανακαλύπτουν ότι το Connected Limited Device Configuration (CLDC) and the profiles τα οποία βασίζονται σε αυτό, δεν είναι απαραίτητο να υποστηρίζουν την ανάγνωση ή την εγγραφή αρχείων. The Generic Connection Framework (GCF) το οποίο ορίζεται από το CLDC παρέχει το βασικό στήριγμα για I/O αρχείων, πρωταρχικά μέσω των *InputConnection, OuputConnection* και *StreamConnection* interfaces, αλλά είναι εξαρτάται από συγκεκριμένες υλοποιήσεις να αποκαλύψουν αυτή την ικανότητα στις εφαρμογές. Αυτός ο περιορισμός δεν είναι κάτι το κακό: Επιτρέπει στο CLDC να έχει φορητότητα σε συσκευές χωρίς σύστημα αρχείων (file system). Εάν ένα σύστημα αρχείων υποστηρίζεται πάντως, θα ήταν καλό να υπάρχει ένας πρότυπος τρόπος για να χρησιμοποιηθεί μέσω του GCF. Αυτό μπορεί να γίνει μέσω του JSR 75.

6.2.2 Ξεκινώντας με τα FileConnection APIs

Στις συσκευές οι οποίες υλοποιούν το JSR 75, μπορούν να ενεργοποιηθούν εφαρμογές που βασίζονται στην J2ME να δημιουργούν, να διαβάζουν και να γράφουν αρχεία και καταλόγους οι οποίοι βρίσκονται σε κινητές συσκευές και εξωτερικές κάρτες μνήμης.

Το JSR 75 παρέχει μερικά πολύ χρήσιμα APIs τα οποία οι προγραμματιστές σε J2ME πραγματικά χρειάζονταν για να εκμεταλλευτούν χαρακτηριστικά τα οποία υπάρχουν σε PDAs, με την μορφή δύο προαιρετικών πακέτων τα οποία επεκτείνουν και βελτιώνουν τα software stacks βασισμένα στο CLDC:

Τα APIs του προαιρετικού πακέτου FileConnection (FCOP) δίνουν στις J2ME συσκευές πρόσβαση στα συστήματα αρχείων τα οποία υπάρχουν στις κινητές συσκευές, και πρόσβαση σε αφαιρούμενα μέσα, όπως κάρτες μνήμης.
Τα APIs του προαιρετικού πακέτου PIM (PIM) δίνουν στις J2ME συσκευές πρόσβαση στην διαχείριση προσωπικών πληροφοριών σε δεδομένα όπως τα βιβλία διευθύνσεων, ημερολόγια, και λίστες υποχρεώσεων.

Είναι σημαντικό να αναφερθεί το FCOP και το PIM είναι ανεξάρτητα το ένα από το άλλο.

Επειδή κάθε συσκευή η οποία καλύπτει τις ελάχιστες απαιτήσεις του CLDC 1.0 μπορεί επίσης και να υποστηρίξει το JSR 75 κι επειδή το Connected Device Configuration (CDC) είναι ένα υπερσύνολο του CLDC, τα APIs του FileConnection, μπορούν να τοποθετηθούν στην κορυφή κάθε CLDC ή CDC profile.

Το FileConnection Optional Package

Όπως αναφέρθηκε και πιο πάνω, για να αποκτήσει κάποιος πρόσβαση στα συστήματα αρχείων τα οποία βρίσκονται στην εσωτερική μνήμη μίας ή συσκευής ή σε μία αφαιρούμενη εξωτερική, όπως οι SmartMedia και CompactFlash κάρτες, το FCOP χρησιμοποιήσει το Generic Connection Framework (GCF) για σύνδεση με το σύστημα αρχείων.

Τα FileConnection APIs

Τα FCOP APIs ορίζονται στο πακέτο javax.microedition.io.file , το οποίο περιέχει δύο interfaces και τρεις κλάσεις:

Πίνακας 6.1 - FileConnection Interfaces

Interface	Περιγραφή
FileConnection	Interface για πρόσβαση σε αρχεία ή καταλόγους
FileSystemListener	Listener interface το οποίο λαμβάνει την ενημερωμένη κατάσταση, όταν προστείθεται ή αφαιρείται ένα file-system root

Πίνακας 6.2 - FileConnection Classes

Class	Περιγραφή
FileSystemRegistry	Central registry για listeners το οποίο πρέπει να προσθέτει ή να αφαιρεί ένα σύστημα αρχείων
ConnectionCloseException	Exception η οποία προκαλείται όταν μία μέθοδος μίας σύνδεσης αρχείου καλείται, αλλά δεν μπορεί να ολοκληρωθεί γιατί η σύνδεση είναι κλειστή.
IllegalModeException	Exception η οποία προκαλείται όταν μία μέθοδος καλείται η οποία απαιτεί ένα συγκεκριμένο security mode, όπως READ ή WRITE, αλλά η σύνδεση που έχει ανοιχτεί δεν είναι σε αυτό το mode

6.2.3 Θέματα Ασφαλείας

Εκτός από το να συμπεριληφθούν όλα τα πακέτα, κλάσεις και interfaces τα οποία ορίζονται στο FCOP, μία σωστή υλοποίηση για το JSR 75 θα έπρεπε να συμπεριλάβει ένα μοντέλο ασφαλείας για την

πρόσβαση στα FileConnection APIs. Πιο συγκεκριμένα:

- Για να προστατευθούνε τα αρχεία και τα δεδομένα του χρήστη από απρόσεκτη ή κακόβουλη πρόσβαση, μία υλοποίηση μπορεί να επιτρέπει την πρόσβαση σε αρχεία τα οποία είναι public και να εμποδίζει την πρόσβαση στα αρχεία τα οποία είναι private ή ευαίσθητα. Η υλοποίηση μπορεί να μην επιτρέπει πρόσβαση στις MIDP RMS βάσεις δεδομένων και δεν θα πρέπει να επιτρέπει την πρόσβαση σε αρχεία ρύθμισης του συστήματος (system configuration files), ή σε αρχεία και καταλόγους τα οποία αφορούν την συσκευή ή το Λειτουργικό Σύστημα, που είναι private ως προς μία άλλη εφαρμογή ή private προς έναν άλλο χρήστη. Σε τέτοιες περιπτώσεις η μέθοδος *Connector.open()* θα πρέπει να προκαλέσει μία *java.lang.SecurityException*.

- Το μοντέλο ασφαλείας θα πρέπει να εφαρμόζεται όταν ανοίγει μία σύνδεση με ένα αρχείο χρησιμοποιόντας την μέθοδο *Connector.open()* και όταν ανοίγει ένα stream για την σύνδεση χρησιμοποιώντας τις μεθόδους *openInputStream()*, *openOutputStream()*, *openDataInputStream()* ή *openDataOutputStream()*.

Ακόμα όμως, εξαρτάται από την πλατφόρμα ή το profile που υπάρχει για το πως θα οριστεί το μοντέλο ασφαλείας. Υπάρχεί μία ειδική περίπτωση: Η ειδική ομάδα προγραμματιστών του JSR 75 πρότεινε μία πρακτική για τα FileConnection APIs όταν το συμπεριλαμβανόμενο profile είναι το MIDP 2.0, η οποία λέει:

- Μη έμπιστα MIDlets τα οποία έχουν πρόσβαση στα προστατευμένα APIs και λειτουργίες των FileConnection APIs θα πρέπει να υπόκεινται σε επιβεβαίωση από το χρήστη.

- Έμπιστα MIDlets πρέπει να προσδιορίουν τα δικαιώματα (permissions) τα οποία εφαρμόζονται στα FileConnection APIs.

6.2.4 Δημιουργώντας Συνδέσεις

Μία εφαρμογή ανοίγει μία σύνδεση χρησιμοποιώντας την μέθοδο *Connector.open()* . Το string που θα χρησιμοποιηθεί σαν παράμετρος θα πρέπει να είναι ένα absolute pathname της μορφής *file://*<*host*>/<*root*>/<*directory*>/<*directory*>/.../<*name*>. Το host element μπορεί να είναι άδειο και συχνά θα είναι, όταν ένα string αναφέρεται σε ένα αρχείο στο loxal host. Το root directory αναταποκρίνεται σε ένα λογικό mount point για μία συγκεκριμένη μονάδα αποθήκευσης. Τα ονόματα του root εξαρτώνται από την συσκευή. Ο παρακάτω πίνακας παρέχει κάποια πραδείγματα root τιμών και πως μπορεί κάποις να τις ανοίξει:

Πίνακας 6.3 - Roots σε κινητές συσκευές

Root Value	Πως να ανοίξει ένα FileConnection
CFCard/	FileConnection fc = (FileConnection) Connector.open("file:///CFCard/");
SDCard/	FileConnection fc = (FileConnection) Connector.open("file:///SDCard/");

Root Value	Πως να ανοίξει ένα FileConnection
MemoryStick/	FileConnection fc = (FileConnection) Connector.open("file:///MemoryStick/");
C:/	FileConnection fc = (FileConnection) Connector.open("file:///C:/");
/	FileConnection fc = (FileConnection) Connector.open("file:////");

Μία σύνδεση σαν τις παραπάνω αναφέρεται σε ένα μόνο αρχείο ή κατάλογο οποιαδήποτε χρονική στιγμή. Ο καλύτερος τρόπος για να γίνει αναφορά σε πολλαπλούς καταλόγους ή αρχεία, είναι να δημιουργηθεί μία ξεχωριστή σύνδεση, η κάθε μία με την χρήση της μεθόδου Connector.open().

Με το που έχει δημιουργηθεί μία σύνδεση σε ένα σύστημα αρχείων μπορούν να γίνουν διάφορα queries με την χρήση των μεθόδων της FileConnection. Μερικά από αυτά είναι:

- Να πάρει κάποιος μία λίστα από αρχεία και καταλόγους που πληρούν κάποια κριτήρια με την χρήση της μεθόδου *list(String filter, boolean includeHidden)*. Στην παράμετρο του string μπορούμε να χρησιμοποιοιήσουμε το σύμβολο * για να προσδιορίσουμε καμία ή περισσότερες εμφανίσεις οποιουδήποτε χαρακτήρα. Στην δεύτερη παράμετρο, μπορούμε να προσδιορίσουμε αν θέλουμε στα αποτέλεσμα και τα κρυφά αρχεία ή όχι.

- Μπορεί κάποιος να δει αν υπάρχει ένα αρχείο ή ένας κατάλογος με την μέθοδο *exists()*.

- Μπορεί κάποιος να δει αν ένα αρχείο ή κατάλογος είναι κρυφός με την μέθοδο *isHidden()*.

- Μπορεί κάποιος να δημιουργήσει ή να διαγράψει ένα αρχείο ή έναν κατάλογο με τις μεθόδους *create()*, *mkdir()*, ή *delete()*.

Για μία λίστα με όλα τις έγκυρες τιμές root στην συσκευή, καλείται η μέθοδος *listRoots()* από την κλάση *FileSystemRegistry*.

Να σημειωθεί πως ένα FileConnection συμπεριφέρεται διαφορετικά από άλλες Generic Connection Framework συνδέσεις με έναν τρόπο. Η Connector.open() μέθοδος μπορεί η επιστροφή της να είναι επιτυχής, χωρίς όμως να αναφέρεται σε κάποιο υπάρχον αρχείο ή κατάλογο. Αυτή η ικανότητα επιτρέπει την δημιουργία νέων αρχείων και καταλόγων. Παρακάτω παρατείθεται η δημιουργία ενός αρχείου, έχοντας μία SDCard στο root.

```
public void createFile() {
   try {
      FileConnection filecon = (FileConnection)
         Connector.open("file:///SDCard/mynewfile.txt");
      // Always check whether the file or directory exists.
      // Create the file if it doesn't exist.
      if(!filecon.exists()) {
         filecon.create();
      }
```

```
      filecon.close();
   } catch(IOException ioe) {
   }
}
```

[22]

Παράδειγμα σύνδεσης σε αρχείο με όνομα data.txt, μέσω του *InputConnection* interface.

```
...
import javax.microedition.io.*;

String url = "file:///data.txt";
InputConnection conn = null;
int mode = Connector.READ_ONLY;

try { to be ported to devices
    conn =(InputConnection) Connector.open( url, mode );
}
catch( IOException ioe ){
    // no file
}
…
```

Αυτός ο κώδικας μπορεί να χρησιμοποιηθεί σε οποιαδήποτε πλατφόρμα η οποία εκθέτει το file system μέσω του GCF, είτε υποστηρίζει το FCOP είτε όχι. Το Connected Device Configuration (CDC) και τα profiles που βασίζονται σε αυτό, υποστηρίζουν αυτή την μέθοδο για να διαβάζονται αρχεία. Μία τέτοια πλατφόρμα, μπορεί επίσης να επιτρέπει την εγγραφή αρχείων εάν κάποιος επιλέξει *Connector.READ_WRITE* ή *Connector.WRITE_ONLY*, επιστρέφοντας ένα *StreamConnection* ή ένα *OutputConnection* σαν κατάλληλο.

Ο δεύτερος τρόπος για να χρησιμοποιήσει κάποιος το FCOP είναι μέσω τους *FileConnection* interface. Πριν το κάνει όμως, θα πρέπει να είναι σίγουρος πως το FCOP είναι διαθέσιμο, ελέγχοντας το με την παρουσία της ιδιότητας *microedition.io.file.FileConnection.version* .

```
...
// Check that the File Connection Optional Package is there

String v = System.getProperty(
    "microedition.io.file.FileConnection.version" );

if( v != null ){
    // FCOP available
} else {
    // FCOP not available
}
```

```
...
```

Η τιμή της ιδιότητας είναι ο αριθμός έκδοσης του FCOP, ο οποίοας είναι "*1.0"* για την πρώτη έκδοση. Εάν η τιμή είναι *null*, τότε δεν υποστηρίζεται.

Όταν το FCOP υποστηρίζεται, όλες οι GFC συνδέσεις γίνονται χρησιμοποιώντας το "*file:"* το πρωτόκολο επιστρέφει ένα instance του *javax.microedition.io.file.FileConnection* . Το *FileConnection* κληρονομεί το *StreamConnection* προσθέτοντας μεθόδους για διαχείριση αρχείων και καταλόγων. Η επιπρόσθετη λειτουργικότητα είναι παρόμοια με εκείνη που είναι διαθέσιμη από την κλάση *java.io.File* στην J2SE.

Επειδή ο αριθμός και τα ονόματα των mounted συστημάτων αρχείων ποικίλει από συσκευή σε συσκευή, παρέχεται μία κλάση η οποία επιτρέπει στις εφαρμογές να βλέπουν τα διαθέσιμα συστήματα αρχείων. Η κλάση *FileSystemRegistry* η οποία ορίζεται στο πακέτο *javax.microedition.io.file* επίσης ενημερώνει τις ενδιαφερόμενες εφαρμογές πότε ο χρήστης εισάγει ή αφαιρεί μία επέκταση κάρτα μνήμης από την συσκευή.

Όταν το URL ενός αρχείου περαστεί στην *Connector.open()* συνήθως αναφέρεται σε ένα υπάρχον αρχείο ή κατάλογο. Για να πάρχουμε την λίστα με αρχεία σε έναν κατάλογο, θα γράφαμε:

```
…
String url = "file:///SDCard";
FileConnection conn = null;

try {
    conn = (FileConnection) Connector.open( url );
    if( conn.isDirectory() ){
        Enumeration names = conn.list();
        while( names.hasMoreElements() ){
            String name = (String) e.nextElement();
            // do something
        }
    } else {
        // not a directory!
    }
}
catch( IOException e ){
    // could not access the URL
}
catch( SecurityException e ){
    // no permission to read the directory
}
...
```

Επίσης το URL μπορεί επίσης να αναφέρεται σε αρχεία ή καταλόγους οι οποίοι δεν υπάρχουν. Για να δημιουργήσουμε ένα αρχείο θα γράφαμε:

```
…
String url = "file:///SDCard/myfile.txt";
FileConnection conn = null;

try {
    conn = (FileConnection) Connector.open( url,
            Connector.WRITE_ONLY );
    if( conn.create() ){ // create the file
        OutputStream out = conn.openOutputStream();
        // now write data to the file
    }

    conn.close();
}
catch( IOException e ){
    // error
}
catch( SecurityException e ){
    // no permission to create/write
}
...
```

Πρέπει να σημειωθεί πως κάθε μέθοδος μέσα στην κλάση FileConnection έχει την πιθαντότητα να προκαλέσει μία SecurityException. Η προδιαγραφή FCOP δεν ορίζει έναν συγκεκριμένο μοντέλο ασφαλείας για να ελέγχεται η πρόσβαση στο σύστημα αρχείων, αφήνοντας την δουλειά στο configuration ή profile που ενσωματώει το FCOP [5].

6.3 Το PIM Optional Package

6.3.1 Εισαγωγή στο PIM Optional Package

Οι περισσότερες, αν όχι όλες οι συσκευές, ασύρματες ή χειροκρατούμενες, οι οποίες κατασκευάζονται σημέρα, έχουν την δυνατότητα να διαχειρίζονται σημαντικές για τον τελικό χρήστη πληροφορίες. Είτε αυτές είναι συναντήσεις στο ημερολόγιό του ή ένα αρχείο με επαφές ή μία λίστα με υποχρεώσεις. Η ικανότητα αυτή, αναφέρεται σαν διαχείριση προσωπικής πληροφορίας (ΔΠΠ) ή PIM για συντομία. Τα δεδομένα PIM αποθηκεύονται στην συσκευή και φυσική ο χρήστης έχει πρόσβαση σε αυτά με μία ή παραπάνω εφαρμογές. Μέχρι το 2003 όμως, τα δεδομένα αυτά δεν ήταν διαθέσιμα στις J2ME εφαρμογές, επειδή κανένα configuration ή profile δεν όριζε τα απαραίτητα προγραμματιστικά interfaces. Το PIM Optional Package όμως καλύπτει αυτό το κενό. Το JSR 75 για την J2ME platform όμως, το οποίο έχει δημιουργηθεί από την Java Community Process, ορίζει το API το οποίο δίνει στις J2ME συσκευές οι οποίες υλοποιούν αυτήν την προδιαγραφή, πρόσβαση στα πρωτογενή δεδομένα PIM στις συσκευές, and to contact and scheduling data in the vCard and vCalendar formats.

6.3.2 Το PIM Optional Package

Ο πρωταρχικός σκοπός του PIM API είναι να δώσει στις J2ME συσκευές, πρόσβαση στα ίδια προσωπικά δεδομένα στα οποία οι εφαρμογές του κινητού, έχουν πρόσβαση. Βασικοί στόχοι αυτού του προαιρετικού πακέτου, είναι:

- Παρέχει πρόσβαση σε προσωπικά δεδομένα σε πρωτογενείς μορφές, τα οποία μπορεί να υπάρχουν στην συσκευή, σε αφαιρούμενα μέσα ή κάπου στο δίκτυο
- Υποστήριξη της εισαγωγής κι εξαγωγής εγγραφών από τον κατάλογο διευθύνσεων σε μορφή vCard, ενώ εγγραφές ημερολογίου και υποχρεώσεων σε μορφή vCalendar
- Δεν επιβάλει απαιτούμενα πεδία και χαρακτηριστικά
- Παρέχει ασφάλεια στην χρήση αυτών των APIs

6.3.3 Δημιουργία συνδέσεων με τα PIM lists

Σαν προαιρετικό πακέτο, το PIMOP δεν είναι ένα απαιτούμενο προγραμματιστικό interface για κανένα J2ME configuration ή profile. Παρ'όλα αυτά, μπορεί να υποστηριχθεί σε κάθε πλατφόρμα J2ME, επειδή εξαρτάται μόνο από τον πυρήνα των κλάσεων ο οποίος ορίζεται από το Connected Limited Device Configuration (CLDC), οι οποίες είναι διαθέσιμες σε όλα τα J2ME profiles. Αν κάποιος προσπαθήσει να γράψει την δική του εφαρμογή η οποία θα εξαρτάται από το PIMOP θα πρέπει να κάνει έναν έλεγχο ώστε να ξέρει αν υποστηρίζεται το PIMOP στην συσκευή του. Ο έλεγχος είναι απλός: αν η ιδιότητα συστήματος `microedition.pim.version` δεν είναι null, τότε το PIMOP είναι διαθέσιμο. Για παράδειγμα:

```
...
// Check that PIM Optional Package is available
String v = System.getProperty( "microedition.pim.version" );
if( v != null ){
    // PIMOP available
} else {
    // PIMOP not available
}
...
```

Η τιμή της ιδιότητας *microedition.pim.version* του PIMOP είναι η τιμής της έκδοσης. Για την πρώτη έκδοση η τιμή είναι "*1.0*".

Στην καρδιά του, το PIMOP είναι όλα όσα χρειάζονται για την διαχείριση λιστών PIM δεδομένων. Το PIMOP ορίζει τρία είδη δεδομένων PIM, στα οποία αναφέρεται συλλογικά σαν *PIM lists*. Το πρώτο είναι το *contact list* (λίστα επαφών), στο οποίο αποθηκεύονται πληροφορίες σχετικά με τις επαφές (ονόματα, διευθύνσεις, αριθμού τηλεφώνου κι άλλα) για άτομα και οργανισμούς. Το δεύτερο είναι το

event list (λίστα σημειώσεων), στο οποίο αποθηκεύονται πληροφορίες σχετικά με σημειώσεις (ραντεβού, υπενθυμίσεις, σημαντικές ημερομηνίες κι άλλα), που έχει ο χρήστης. Το τρίτο είναι το *todo list* (λίστα υποχρεώσεων), στο οποίο αποθηκεύονται οι εργασίες τις οποίες πρέπει να κάνει ο χρήστης. Δεν υποστηρίζονται σε όλες τις συσκευές και τα τρία ήδη δεδομένων PIM, αλλά τουλάχιστον ένα από αυτά θα πρέπει να είναι διαθέσιμο αν ο δημιουργός της εφαρμογής λέει πως υποστηρίζει το PIMOP.

Όλες οι κλάσεις και τα interfaces τα οποία ορίζονται από το PIMOP είναι στο πακέτο *javax.microedition.pim*. Μπορεί κάποιος να έχεις πρόσβαση τα δεδομένα PIM με ένα instance της κλάσης *javax.microedition.pim.PIM* χρησιμοποιώντας την static μέθοδο *getInstance()*:

```
...
import javax.microedition.pim.*;
PIM singleton = PIM.getInstance();
...
```

Ύστερα γίνεται κλήση της μεθόδου *openPIMList()* για να έχουμε πρόσβαση σε μία PIM list. Όλες οι PIM lists αντιπροσωπεύονται από την κλάση *PIMList* και τις τρεις υποκλάσεις: *ContactList, EventList,* και *ToDoList*. Η πρώτη παράμετρος στην *openPIMList()* ορίζει το είδος της PIM list που θέλουμε να μας επιστραφεί και πρέπει να είναι μία από τις τιμές *PIM.CONTACT_LIST, PIM.EVENT_LIST,* ή *PIM.TODO_LIST*. Η δεύτερη παράμετρος ορίζει το είδος τροποποίησης. *PIM.READ_ONLY, PIM.READ_WRITE* ή *PIM.WRITE_ONLY*. Υπάρχει και μία προαιρετική τρίτη παράμετρος, το όνομα της επιθυμητής PIM list. Αν το όνομα δεν συγκεκριμενοποιείται, τότε η μέθοδος επιστρέφει την default PIM list για τον δοσμένο τύπο.

```
...
PIM singleton = PIM.getInstance();
ContactList cl = null;

try {
    cl = (ContactList)singleton.openPIMList(PIM.CONTACT_LIST,
                                            PIM.READ_ONLY );
    // use the contact list
}
catch( PIMException ){
    // no contact list available!
}
catch( SecurityException ){
    // the application is not allowed to access the list
}
...
```

Εάν η επιθυμητή λίστα δεν είναι διαθέσιμη, τότε η *openPIMList()* ρίχνει (throws) μία *PIMException*. Να σημειωθεί επίσης, ότι αυτή η μέθοδος μπορεί να ρίξει και μία *java.lang.SecurityException*. Το περιβάλλον της J2ME μπορεί να θέσει περιορισμούς στο ποιες εφαρμογές μπορούν να διαβάσουν ή να

γράψουν δεδομένα PIM, χρησιμοποιώντας οποιοδήποτε μοντέλο ασφαλείας ταιριάζει. Ένα MIDP 2.0 περιβάλλον για παράδειγμα, μπορεί επιτρέψει μόνο σε "έμπιστα" MIDlets να έχουν πρόσβαση στο contanct list της συσκευής.

Το PIM optional package API ορίζει τρεις τύπους PIM δεδομένων, γνωστών και ως PIM lists:

- Contact list η οποία περιέχει ονόματα, διευθύνσεις, τηλέφωνα και άλλες επαγγελματικές και προσωπικές πληροφορίες
- Event list η οποία περιέχει συναντήσει, υπενθυμίσεις και άλλα στοιχεία σχετικά με το ημερολόγιο
- To-do list η οποία περιέχει εγγραφές με τις υποχρεώσεις τις οποίες έχει ο χρήστης

Μία PIM list είναι ένα container για έναν αριθμό εγγραφών ή στοιχείων PIM. Με χρήση της μεθόδου *items()* οποιουδήποτε ονόματος λίστας, παίρνουμε μία απαρίθμηση (enumeration) των στοιχείων PIM. Για παράδειγμα:

```
...
import java.microedition.pim.*;
import java.util.*;
ContactList list = ... // a contact list
try {
    Enumeration enum = list.items();
    while( enum.hasMoreElements() ){
        Contact contact = (Contact) enum.nextElement();
        // do something with the contact
    }
}
catch( PIMException e ){
    // an error occurred
}
catch( SecurityException e ){
    // can't read this list
}
...
```

Μπορούμε επίσης να απαριθμήσουμε στοιχεία τα οποία υπόκεινται σε συγκεκριμένα κριτήρια ή ανήκουν σε συγκεκριμένες κατηγορίες (ομάδες ή στοιχεία). Άλλες μέθοδοι επιτρέπουν την προσθήκη ή αφαίρεση στοιχείων από την λίστα και την απόκτηση πληροφοριών από τα πεδία δεδομένων των στοιχείων.

Τα συγκεκριμένα πεδία τα οποία είναι διαθέσιμα σε ένα στοιχείο PIM μπορεί να ποικίλει ανάλογα με την συσκευή και τον τύπο της λίστας. Η προδιαγραφή του PIMOP ορίζει πρότυπα πεδία για κάθε είδος στοιχείου λίστας. Για τα πιο πολλά μέρη ένα υποσύνολο των πεδίων ορίζεται στις προδιαγραφές των

vCard και vCalendar (πρότυπα internet για την ανταλλαγή επαφών και πληροφοριών ημερολογίου). Μία ιδιαίτερη υλοποίηση μπορεί να υποστηρίξει μόνο συγκεκριμένα πεδία, οπότε είναι σημαντικό το να ελέγξουμε την υλοποίηση ώστε να δούμε αν το συγκεκριμένο πεδίο είναι διαθέσιμο, με την κλήση της μεθόδου *isSupportedField()*. Μία *UnsupportedFieldException* προκαλείται όποτε μία εφαρμογή προσπαθεί να χειριστεί ένα πεδίο του οποίου η υλοποίηση δεν υποστηρίζεται.

Η προδιαγραφή PIMOP επίσης ορίζει πρότυπους τύπους δεδομένων για τα πεδία επειδή μπορεί κάποια από αυτά να αποθηκεύουν strings, άλλα ημερομηνίες και διάφορα άλλα. Ο τύπος του πεδίου αποφασίζει ποιες μέθοδοι χρησιμοποιούνται για να χειριστούν τις τιμές τους. Για παράδειγμα, το πεδίο *TEL* ενός Contact (ο αριθμός τηλεφώνου της επαφής) είναι ένα string το οποίου το παίρνουμε με την χρήση της μεθόδου *getString()*:

```
...
Contact contact = ...;
String tel = contact.getString( Contact.TEL, 0 );
...
```

Παρ'όλα αυτά όμως, το πεδίο *BIRTHDAY* είναι τύπου long και παίρνουμε τα στοιχεία του με την χρήση της μεθόδου *getDate()*:

```
...
Contact contact = ...;
long bday = contact.getDate( Contact.BIRTHDAY, 0 );
...
```

Να σημειωθεί πως κάθε πεδίο μπορεί να αποθηκεύσει πολλαπλές τιμές, όπως κάνει ένα Vector. Η δεύτερη παράμετρος σε μεθόδους όπως η *getString()* είναι ένας δείκτης (index) που αναγνωρίζει ποια από τις τιμές θελουμε [4].

6.3.4 Τα PIM APIS

Τα PIM APIs ορίζονται στο πακέτο *javax.microedition.io.pim*. Αυτό το πακέτο περιέχει 8 interfaces κι 6 classes, περιλαμβάνοντας και 4 τύπους εξαιρέσεων

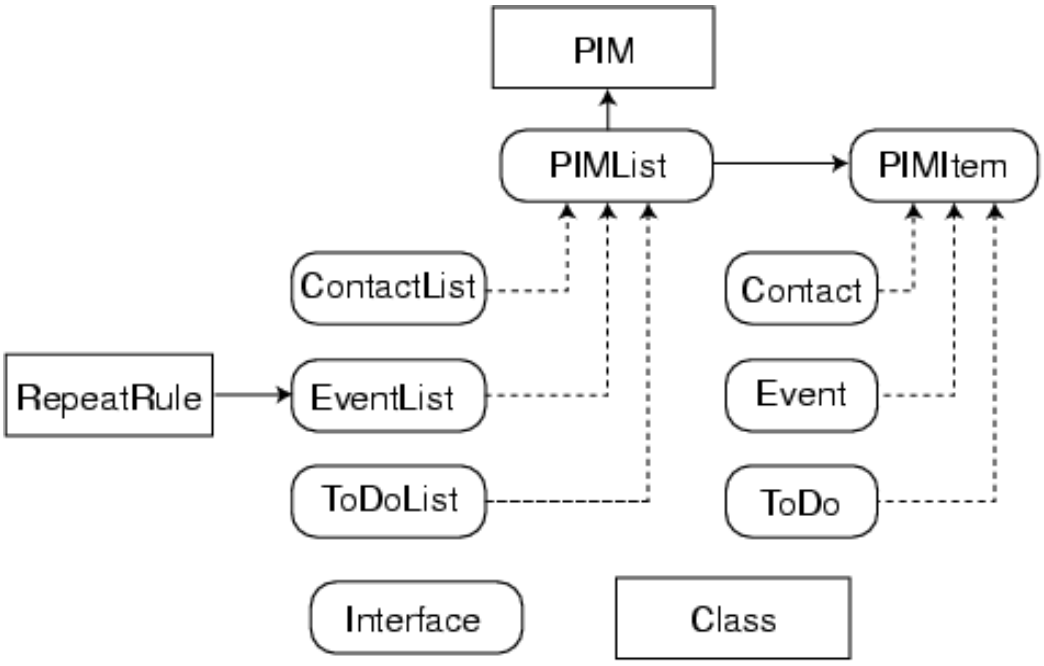
Πίνακας 6.4 - PIM Interfaces

Interfaces	Description
`PIMItem`	Το κοινό superinterface ενός item το οποίο θα αποθηκευθεί σε μία PIM list, όπου item είναι μία συλλογή από δεδομένα για μία εγγραφή PIM: ένα `Contact`, `Event` ή `ToDo`
`PIMList`	Το κοινό superinterface των `ContactList`, `EventList`, και `ToDoList`, κάθε ένα από τα οποία μπορεί να περιέχει μηδέν ή περισσότερα `PIMItems`

Interfaces	Description
`Contact`	Μία μοναδική εγγραφή σε μία βάση δεδομένων για επαφές. Τα static πεδία σε αυτό το interface είναι υποσύνολα των πεδίων τα οποία ορίζονται από την προδιαγραφή vCard
`ContactList`	Μία λίστα από `Contact` items
`Event`	Μία μοναδική εγγραφή για την event βάση δεδομένων
`EventList`	Μία λίστα από `Event` items
`ToDo`	Ένα μοναδικό item σε μία a to-do βάση δεδομένων
`ToDoList`	Μία λίστα από `ToDo` items

Πίνακας 6.5 - PIM Classes

Classes	Description
`PIM`	Παρέχει μία συλλογή από static μεθόδους για να βρούμε πληροφορίες και να έχουμε πρόσβαση στις `PIMLists`
`RepeatRule`	Η περιγραφή ενός προτύπου επανάληψης για ένα `Event` item, για να προσδιορίσει πότε το event θα συμβεί. Τα static πεδία σε αυτή την class είναι ένα υποσύνολο των δυνατοτήτων του `RRULE` πεδίου στο `VEVENT`, το οποίο ορίζεται από την προδιαγραφή vCalendar 1.0
`FieldEmptyException`	Ρϊχνεται όταν γίνεται μία προσπάθεια για πρόσβαση σε ένα πεδίο το οποίο δεν έχει κάποια δεδομένα
`FieldFullException`	Ρίχνεται όταν γίνεται μία προσπάθεια για εισαγωγή τιμής σε κάπιο πεδίο, όταν αυτό είναι ήδη γεμάτο
`PIMException`	Ρίχνεται από τις PIM classes
`UnsupportedException`	Ρίχνεται από το αναφερόμενο πεδίο δεν υποστηρίζεται από την PIM list



Εικόνα 6.1 - Η ιεραρχία του PIM

6.4 Ανακεφαλαίωση

Σε αυτό το κεφάλαιο με την χρήση διαφόρων παραδειγμάτων, ο αναγνώστης κατεννόησε πως μπορεί να έχει πρόσβαση στο file system μίας κινητής συσκευής, να δημιουργήσει διάφορα αρχεία, και να αποθηκεύσει δεδομένα σε αυτά. Ακόμη είδε πως είναι τα συστατικά μέρη του PIM και πως μπορεί να έχει πρόσβαση και έλεγχο των δεδομένων του.

Στο επόμενο κεφάλαιο, θα δούμε το μεγαλύτερο πρόβλημα της J2ME, το οποίο βασανίζει για χρόνια ολόκληρα τους προγραμματιστές και συνεπώς τους χρήστες. Το όνομα αυτού; Κατακερματισμός.

Κεφάλαιο 7ο

7.1 Εισαγωγή

Σε αυτό το κεφάλαιο θα δούμε το φαινόμενο του Κατακερματισμού, το οποίο βασανίζει τους προγραμματιστές και τους χρήστες. Παρακάτω αναλύονται τα είδη του και οι τρόποι επίλυσής του.

7.2 Κατακερματισμός

7.2.1 Εισαγωγή στον κατακερματισμό

Η Java ME platform χρησιμοποιείται σε εκατομμύρια συσκευές, υποστηρίζεται από πάρα πολλές εταιρείες και χρησιμοποιείται παγκοσμίως. Όμως όλη αυτή η επιτυχία, δεν έχει έρθει χωρίς μερικές προκλήσεις. Ένα συνηθισμένο θέμα συζητήσεων στην βιομηχανία ανάπτυξης λογισμικού και στους προγραμματιστές, είναι ο κατακερματισμός. Με τον όρο αυτό, εννοούμε την generic μόδα σε κάθε περίπτωση στην οποία η πρόταση Write Once, Run Anywhere™ αποτυγχάνει στο να παρέχει μία ενιαία πλατφόρμα εκτέλεσης σε όλες τις συσκευές και τα συστήματα. Το πιο συνηθισμένο παράδειγμα κατακερματισμού είναι όταν μία εφαρμογή δεν τρέχει σε όλες τις υποστηριζόμενες συσκευές ή όταν μία εφαρμογή συμπεριφέρεται διαφορετικά κατά την εκτέλεσή της, ανάλογα με την συσκευή.

Οι προγραμματιστές, οι πάροχοι και διανομείς, οι κατασκευαστές και πολλοί άλλοι επιρεάζονται από αυτό το γεγονός. Όταν μία εφαρμογή σε Java γράφεται, θα πρέπει να τρέχει σε όλες τις υποστηριζόμενες συσκευές, χωρίς επιπρόσθετες τροποποιήσεις και δοκιμές. Παρ'όλα αυτά, κάτι τέτοιο δεν συμβαίνει και πολλές φορές θα πρέπει η εφαρμογή να τροποποιηθεί και να ελεγχθεί για κάθε μία συσκευή. Δεδομένου του τεράστιου αριθμού των συσκευών, κάτι τέτοιο είναι πολύπλοκο και κοστίζει υπερβολικά.

Ο κατακερματισμός είναι ένα πολύπλοκο θέμα το οποία πηγάζει από τις τροποποιήσεις που κάνουν οι κατασκευαστές στις πλατφόρμες που χρησιμοποιούν, ώστε να διαφέρουν σε καινοτομία από τις υπόλοιπες στην αγορά.

7.2.2 Τύποι κατακερματισμού

Στην ανάλυση του κατακερματισμού, είναι δυνατόν να προσδιοριστούν πολλές διαφορετικές μορφές κατακερματισμού, ανάλογα με την κύρια πηγή των προβλημάτων διαλειτουργικότητας. Τα τρία πιο κοινά είδη κατακερματισμού είναι:

- Κατακερματισμός πλατφόρμας
- Κατακερματισμός υλοποίησης
- Κατακερματισμός επιπέδου συσκευής

Κατακερματισμός πλατφόρμας

Κατακερματισμός πλατφόρμας είναι ένας γενικός όρος που αναφέρεται στις συσκευές που υποστηρίζουν ένα διαφορετικό σύνολο από APIs της Java τεχνολογίας. Αυτό το είδος του κατακερματισμού που αναφέρεται επίσης ως API κατακερματισμός ή επιπέδου API κατακερματισμός.

Υπάρχουν πολλαπλοί λόγοι για τον κατακερματισμό πλατφόρμας. Η ύπαρξη των συσκευών από διάφορες φάσεις της εξελιξης την πλατφόρμας Java ME σημαίνει ότι οι προγραμματιστές και οι φορείς παροχής περιεχομένου, αναπόφευκτα να λάβουν υπόψην τους την παρουσία συσκευών με διαφορετικά API. Άλλοι λόγοι για την κοινή πλατφόρμα κατακερματισμό περιλαμβάνουν την παρουσία:

- Κατασκευή ειδικών APIs και επεκτάσεων.
- Προαιρετικά ή «υπό όρους υποχρεωτικά συστατικά, όπως το Bluetooth ή των υπηρεσιών εντοπισμού θέσης.
- Διαφορετικές εκδόσεις του ίδιου API σε διαφορετικές συσκευές.
- Νέες προδιαγραφές για την πλατφόρμα Java ME οι οποίες δεν είναι ακόμα μέρος κάποιας "ομπρέλας" πρότυπου ακόμα.

Κατακερματισμός υλοποίησης

Η δεύτερη μορφή του κατακερματισμού είναι ο κατακερματισμός υλοποίησης. Αυτό το είδος κατακερματισμό αναφέρεται στην γενική κατάσταση στην οποία οι συσκευές για την πλατφόρμα Java ME υλοποιούν τα ίδια APIs διαφορετικά, συνήθως τυχαία και όχι εσκεμμένα. Ο κατακερματισμός υλοποίησης μερικές φορές αναφέρεται και ως κατακερματισμός ποιότητα ή κατακερματισμός υλοποίησης επιπέδου. Κοινές πηγές του κατακερματισμού υλοποίησης περιλαμβάνουν τα ακόλουθα:

- Ελλιπείς ή ασαφείς προδιαγραφές. Οι προδιαγραφές δεν είναι πάντα πλήρεις ή αρκετά λεπτομερείς ώστε να εγγυηθούν την πλήρη συμβατότητα του επιπέδου υλοποίησης. Σε πολλές περιπτώσεις, οι προδιαγραφές είναι ασαφείς, αφήνοντας πολύ μεγάλα περιθώρια ερμηνείας κατά την υλοποίησή τους.

- Ανεπαρκής TCK κάλυψη δοκιμών. Τα Technology Compatibility Kits (TCKs) δεν παρέχουν πάντοτε αρκετά σενάρια δοκιμών για να εγγυηθούν 100% συμβατότητα όλων των συσκευών οι οποίες υλοποιούν μία προδιαγραφή.

- Η ανεπαρκής αξιολόγηση της ποιότητας / ποιοτικού ελέγχου. Ο κατακερματισμός υλοποίησης προκαλείται επίσης από κοινά σφάλματα αλλά και από άλλα θεμάτα ποιότητας. Από την στιγμή που τα TCKs έχουν δοκιμές μόνο για τον έλεγχο συμβατότητας, εκτελώντας ένα TCK με επιτυχία δεν εγγυάται από μόνο του υψηλής ποιότητας εκτέλεση. Αντίθετα, εκτός από τον έλεγχο συμβατότητας, πρόσθετες δοκιμές ποιότητας είναι πάντα αναγκαίες για τη διασφάλιση υψηλής ποιότητας εκτέλεσης.

- Ανεπαρκείς δοκιμές διαλειτουργικότητας. Προβλήματα συμβατότητας συχνά προκαλούνται από τις διαφορές υλοποίησης και του επιπέδου των συσκευών ανάμεσα στις συσκευές που είναι

πέραν των προδιαγραφών της βιομηχανίας (π.χ., ρυθμίσεις ιδιόκτητου δικτύου, διακυμάνσεις στις πολιτικές ασφαλείας, τις νομικές απαιτήσεις σε διαφορετικές χώρες, κ.λπ.). Επιπρόσθετες δοκιμές διαλειτουργικότητας είναι συνήθως απαραίτητες για τη διασφάλιση της συμβατότητας μεταξύ των συσκευών με διαφορετικά χαρακτηριστικά.

- Χρόνος διάθεσης στην αγορά. Μια τυπική γενική πηγή κατακερματισμού επιπέδου υλοποίησης είναι ο χρόνος έως τη διάθεση στην αγορά. Οι περισσότεροι πωλητές είναι υπό συνεχή πίεση να θέσουν στην αγορά τα νέα προϊόντα και τα νέα χαρακτηρτιστικά. Σε ορισμένες περιπτώσεις, απλώς δεν υπάρχει αρκετός χρόνος για να εξασφαλιστεί ότι όλα τα χαρακτηριστικά υλοποιούνται σωστά.

Κατακερματισμός επιπέδου συσκευής

Η τρίτη αυτή μορφή του κατακερματισμού προκύπτει από τα διαφορετικά χαρακτηριστικά των συσκευών που υποστηρίζουν την πλατφόρμα Java ME. Υπάρχουν πολλοί παράγοντες που προκαλούν την συσκευή σε επίπεδο κατακερματισμού:

- Διαφορετικά μεγέθη οθόνης και διαφορετικά χρώματα
- Διαφορετικές διατάξεις πληκτρολογίου και υποστηριζόμενα πλήκτρα και/ή λειτουργίες δεικτών
- Διαφορετικές ταχύτητες CPU
- Διαφορετικά μεγέθη RAM και διαφορετικά μεγέθη μνήμης αποθήκευσης
- Η παρουσία ή η απουσία ενός file system
- Υποστήριξη για εξωτερική και/ή αφαιρούμενη κάρτας μνήμης
- Διαφορετικές υποστηριζόμενες επιλογές για αποστολή μηνυμάτων και δικτύου
- Διαφορετικές δυνατότητες μέσων όπως η παρουσία κάμερας, καταγραφής ήχου ή εικόνας και η ύπαρξη κωδικοποιητών
- 2D/3D επιταχυντές γραφικών, βιβλιοθήκες για αριθμούς κινητής υποδιαστολής

Εικόνα 7.1 - Κινητές Συσκευές

Οι διαφορές επιπέδου συσκευής μπορούν να κάνουν τις εφαρμογές να συμπεριφέρονται με μη συμβατό τρόπο, ακόμη και αν η υποκείμενες υλοποιήσεις της πλατφόρμα θα ήταν διαφορετικά 100% συμβατές. Οι διαφορές επιπέδου συσκευής μπορεί επίσης να προκαλέσει σημαντική υποβάθμιση στην ευχρηστία της εφαρμογής. Σκεφτείτε, για παράδειγμα, τους τρεις διαφορετικούς τύπους των κινητών συσκευών που απεικονίζεται στην παραπάνω εικόνα. Ακόμη και αν οι εν λόγω συσκευές έχουν ακριβώς τα ίδια χαρακτηριστικά υλικού και το ίδιο λογισμικό πλατφόρμας, τα διαφορετικά μεγέθη οθόνης και η χρήση των μοντέλων των συσκευών αυτών καθιστά τη συμπεριφορά και τη χρηστικότητα των εφαρμογών μία πρόκληση.

Το παραπάνω παράδειγμα είναι μόνο η κορυφή τους παγόβουνου όσον αφορά τις προκλήσεις που υπάρχοτυν από τις επιπέδου συσκευής διαφορές. Στην πραγματικότητα, είναι πολύ δύσκολο να οικοδομηθεί μια πλατφόρμα λογισμικού η οποία θα λειτουργούσε καλά σε μια μεγάλη ποικιλία διαφόρων τύπων συσκευών με τόσο πολλές διαφορετικές παραλλαγές στα χαρακτηριστικά της συσκευής.

Άλλοι τύποι κατακερματισμού

Επιπρόσθετα στις τρεις πιο κοινές μορφές του κατακερματισμού, υπάρχουν και κάποιες άλλες ακόμη, οι οποίες κι αυτές μπορούν να προκαλέσουν προβλήματα συμβατότητας.

Πρότυπα κατακερματισμού

Για ιστορικούς λόγους, υπάρχουν διαφορετικά πρότυπα για τα διάφορα μέρη του κόσμου. Για παράδειγμα, από νωρίς στην ιστορία της πλατφόρμας Java ME, ένα διαφορετικό πρότυπο, ονομζόταν iAppli, αναπτύχθηκε για την ιαπωνική αγορά. Ομοίως, στην αγορά της Κορέας αναπτύχθηκε το πρότυπο που χρησιμοποιείται σήμερα, το WIPI.

Κατακερματισμός ανάπτυξης υποδομής και πολιτικής

Σε κάποιο βαθμό, ο κατακερματισμός προκαλείται από τις διακυμάνσεις στο υποκείμενο δίκτυο και στον κύριο κορμό της υποδομής. Για παράδειγμα, διαφορετικές πολιτικές ασφάλειας που εγκρίθηκαν από τους διαχειριστές δικτύων μπορεί μερικές φορές να αποτρέψουν μια Java εφαρμογή να λειτουργήσει σωστά σε ορισμένα δίκτυα, όπως όταν η πρόσβαση σε ορισμένα πρωτόκολλα δικτύου απαγορεύεται. Ακόμη, αν το bandwidth ενός δικτύου έχει περιορισμούς εφαρμογή θα συμπεριφερθεί διαφορετικά κατά την εκτέλεσή της, κλπ. Αυτού του είδους τα θέματα πολιτικής είναι πέρα από τα ζητήματα τα οποία μπορούν να αντιμετωπιστούν από τα παγκόσμια πρότυπα.

Localization κατακερματισμός

Στις μέρες μας, οι εφαρμογές διαδίδονται σε ολόκληρο τον κόσμο πολύ εύκολα. Έτσι μία εφαρμογή η οποία απευθείνεται σε χρήστες διαφορετικών χωρών ή ακόμα και ηπείρων, θα πρέπει να είναι μπορεί να αναπεξέλθει στην πολυγλωσσικότητα, αλλά και στις άλλες ιδιαιτερότητες του κάθε λαού. Η

διαδικασία της προσαρμογής μία εφαρμογής σε διαφορετικές γλώσσες και στις ιδιαιτερότητες του κάθε λαού, είναι γνωστή ως localization. Το localization είναι ένα σύνθετο θέμα που δεν περιορίζεται μόνο στη γλώσσα του κειμένου (συμβολοσειρές) που παρουσιάζονται στο χρήστη. Ορισμένες γλώσσες, όπως τα ρωσικά, κινέζικα, ιαπωνικά έχουν ένα πολύ διαφορετικό σύνολο χαρακτήρων. Οι συνθήκες, π.χ., για ορισμένες μορφές, νομίσματα, ημερομηνίες και ώρες μπορεί να ποικίλλει σημαντικά από χώρα σε χώρα. Τα προβλήματα κατακερματισμού στον τομέα αυτό είναι κάπως παρόμοια με εκείνα του κατακερματισμού επιπέδου συσκευής (π.χ., το μέγεθος της οθόνης). Χωρίς το localization η τοποθέτηση εικονιδίως, συμβόλων και διαφόρων άλλων, μπορεί να έχουν σημαντικό αντίκτυπο στη χρηστικότητα της εφαρμογής. Ακόμα κι αν υπήρχε ένα μοναδικό παγκόσμιο πρότυπο για την αναάπτυξη μίας εφαρμογής, το πραγματικό Write Once, Run Anywhere™ δεν θα είναι, παρά μόνο μία πλάνη, αν τα θέματα τα οποία αφορούν το localization δεν ληφθούν υπόψην, κατά την διάρκεια ανάπτυξης της εφαρμογής.

7.2.3 Αντιμετώπιση του κατακερματισμού

Πολλά από τα προβλήματα κατακερματισμού τα οποία έχουν παρατηρηθεί στην βιομηχανία της Java ME τεχνολογίας ξεκινούν από το γεγονός ότι είναι διαδεδομένη και δημοφιλής ανάμεσα στους κατασκευαστές των διαφόρων τύπων συκευών.

Η Sun συνεργάζεται με μεγάλα ονόματα κατασκευαστών συσκευών, για να μπορέσει να επιλύσει αυτό το μεγάλο πρόβλημα. Οι λύσεις που υπάρχουν είναι μέσω:

- της JCP.
- δοκιμών και επαλήθευσης.
- βελτιστοποιημένων υλοποιήσεων.
- ανοιχτού κώδικα.
- της υποστήριξης βελτιωμένων εργαλείων.
- εκπαίδευσης των προγραμματιστών.
- συνεχής βελτίωση του προγράμματος της JCP.

Αντιμετώπιση του κατακερματισμού μέσω της JCP

Ένας από τους πιο σημαντικούς τρόπους για την αντιμετώπιση του κατακερματισμού στην αγορά της Java ME τεχνολογίας είναι η δημιουργία ενός προγράμματος με πρότυπα και πρωτοβουλίες, από την JCP, το οποίο θα καθορίζει μια κοινή βασική πλατφόρμα για το σύνολο της βιομηχανίας. Τα καλύτερα παραδείγματα αυτών των δραστηριοτήτων κατά τη διάρκεια των τελευταίων ετών στο χώρο της κινητών συσκευών είναι η πρωτοβουλία για την Java Tecnhology for the Wireless Industry (JTWI, JSR 185) και η πιο πρόσφατη Mobile Service Architecture (MSA, JSR 248 και JSR 249).

Οι JTWI και MSA ορίζουν πλατφόρμες οι οποίες προορίζονται να χρησιμοποιηθούν ως κοινά πρότυπα σε ολόκληρη την βιομηχανία κινητών συσκευών. Η JTWI είναι εξαιρετικά δημοφιλής, με πολλές εκατοντάδες εκατομμύρια να την χρησιμοποιούν, κάτι το οποίο όμως πλέον αλλάζει από την πιο πρόσφατη MSA.

Το πρόγραμμα προτύπων της JCP μπορεί να μειώσει τον κατακερματισμό με δύο τρόπους:

- **Μείωση του κατακερματισμού πλατφόρμας:** Πρώτον, με τον καθορισμό ενός κοινού συνόλου APIs το οποίο όλες οι συσκευές πρέπει να περιλαμβάνυν. Έτσι θεσπίζεται μία πλατφόρμα, η οποία έχει το ελάχιστο επίπεδο λειτουργικότητας το οποίο όλες οι συσκευές σε όλη τη βιομηχανία θα πρέπει να καλύπτουν. Επίσης, μπορεί να μειωθεί με την ελαχιστοποίηση των προαιρετικών και «υπό όρους υποχρεωτικών» χαρακτηριστικών.

- **Μείωση του κατακερματισμού υλοποίησης:** Με τον καθορισμό πρόσθετων διευκρινίσεων και απαιτήσεων που διευκρινίζουν την συμπεριφορά των προδιαγραφών. Αυτές οι προδιαγραφές εξαλείφουν σφάλματα, ασάφειες, ασαφή στοιχεία και άλλες πιθανές παρανοήσεις τι οποίες μπορεί οι προδιαγραφές να έχουν.

Αντιμετώπιση του καταμερματισμού μέσω δοκιμών κι επαλήθευσης

Όπως ήδη αναφέρθηκε, τα TCKs από μόνα τους δεν μπορούν να εγγυηθούν την υψηλή ποιότητα της υλοποίησης μίας προδιαγραφής. Από την στιγμή που τα TCKs επικεντρώνονται μόνο στη συμβατότητα, πρόσθετες δοκιμές είναι απαραίτητες για να βοηθήσουν στον να επικυρωθεί η πληρότητα και η απόδοση όλων των λειτουργιών που προσδιορίζονται. Σε γενικές γραμμές, οι δοκιμές πάνω στην λειτουργικότητα, έλεγχος ποιότητας, η συγκριτική αξιολόγηση των επιδόσεων και άλλα σημαντικά ζητήματα πρέπει να αντιμετωπιστούν μέσω πρόσθετων προσπαθειών εκτός από τα TCKs.

Η Sun έχει πάρει κάποιες πρωτοβουλίες για αυτό τον σκοπό. Αυτές είναι:

- Java Device Test Suite java.sun.com/products/javadevice
- Unified Testing Iniative http://www.javaverified.com/about_uti.jsp
- Java Verified Program http://javaverified.com

Αντιμετώπιση του κατακερματισμού μέσω βελτιστοποιημένων υλοποιήσεων

Στις πρώτες ημέρες εξέλιξης της Java ME πλατφόρμας, η κάθε εταιρεία του διατηρούσε τις δικές της υλοποίησης της Java platform stack συμπεριλαμβανομένων των υλοποιήσεων όλων των βασικών προδιαγραφών. Αυτή η ποικιλομορφία στην υλοποίησης (βασικά: ξεχωριστός πηγαίος κώδικας για κάθε εταιρεία) συνέβαλε στις αποκλίσεις και τα σφάλματα που βλέπουμε ακόμα και σήμερα.

Ένας τρόπος για την αντιμετώπιση του κατακερματισμού υλοποίησης είναι η χρήση ενός ενιαίου πηγαίου κώδικα, δηλαδή, ανάμεσα στους διάφορους κατασκευαστές. Για το σκοπό αυτό, η Sun προσφέρει προσαρμοσμένες υλοποιήσεις όλων των βασικών προτύπων της Java ME με την μορφή βελτιστοποιημένων υλοποιήσεων. Οι βελτιστοποιημένες υλοποιήσεις περιλαμβάνουν διάφορα χαρακτηριστικά τα οποία εστιάζονται ειδικά σε παράγοντες της ποιότητας των προϊόντων όπως η απόδοση, η φορητότητα, η επεκτασιμότητα, η πλήρης συμβατότητα, και η ευκολότερη συντήρηση κώδικα. Εκτός από τις βελτιστοποιημένες υλοποιήσεις, η Sun προσφέρει επίσης υπηρεσίες τεχνικής υποστήριξης οι οποίες είναι διαθέσιμες σε περίπτωση που επιπλέον παραμετροποίηση είναι ακόμη επιθυμητή από τον πελάτη.

Αντιμετώπιση του κατακερματισμού μέσω του ανοιχτού κώδικα

Ένας σημαντικός τρόπος για να μειωθεί σημαντικά ο κατακερματισμός είναι ο ανοιχτός κώδικας. Με το άνοιγμα του πηγαίου κώδικα της υλοποίησης Java platfrom στην κοινότητα ανάπτυξης, είναι δυνατό να ενθαρρυνθεί το σύνολο της βιομηχανίας να υιοθετήσει μία κοινή υλοποίηση, η οποία να διατηρείται σε συνεργασία με τη βιομηχανία.

Αντιμετώπιση του κατακερματισμού μέσω της υποστήριξης με βελτιωμένα εργαλεία

Ακόμη και οι υλοποιήσεις της Java ME πλατφόρμας σε κάθε συσκευή ήταν πανομοιότυπη, και πάλι θα παραμέναν διαφορές μεταξύ των διαφόρων συσκευών (το μέγεθος της οθόνης, η διάταξη των κουμπιών κλπ.). Ως εκ τούτου, ο προγραμματιστής πρέπει να αποδεχθεί ότι ένα binary αρχείο δεν θα επαρκεί για την κάλυψη των αναγκών όλων των χρηστών. Με αυτό κατά νου, αρκετά εργαλεία έχουν ασχοληθεί με το θέμα αυτό με την προσθήκη υποστήριξης για preprocessing.

Η preprocessing προσέγγιση επιτρέπει στους προγραμματιστές να δημιουργήσουν πολλές εκδόσεις της εφαρμογής όλες, από ένα πηγαίο αρχείο, χωρίς να απαιτεί ένα τεράστιο δυαδικό αρχείο γεμάτο δομές επιλογής. Αντ 'αυτού, χρησιμοποιώντας κάποια εργαλεία, οι προγραμματιστές μπορούν να δημιουργήσουν πολλά σενάρια τα οποία χρειάζονται, μέσα σε ένα μόνο κύριο αρχικό αρχείο. Οι δομές επιλογών μπορούν να αντιμετωπίσουν κάθε είδους διαφοροποίηση συμπεριλαμβάνομένου:

- Χρήση ή μή ενός API το οποίο εξαρτάται από την συσκευή
- UI tweaks τα οποία εξαρτώνται από την φυσική μορφή της συσκευής
- Βελτιστοποιώντας για συγκεκριμένο επεξεργαστή ή περιορισμούς μνήμης
- Λύσεις για γνωστά θέματα και bugs
- Επίλυση του localization

Αντιμετώπιση του κατακερματισμού μέσω της εκπαίδευσης των προγραμματιστών

Αυτό είναι ένα θέμα εκπαίδευσης των προγραμματιστών. Να έχουν ρεαλιστικές προσδοκίες σχετικά με τη φορητότητα. Να γνωρίζουν τι ακριβώς χρειάζεται για να γράψει κάποιος μια πραγματικά φορητή εφαρμογή. Να γνωρίζουν σχετικά με τις τεχνικές, τις συμβάσεις και τα πρότυπα σχεδιασμού τα οποία μπορεί να χρησιμοποιηθούν, όπως για παράδειγμα να λαμβάνουν υπόψην τα διαφορετικά μεγέθη οθόνης , τις διαφορετικές συμπεριφορές που προκαλούνται από διακυμάνσεις του localization ή διαφορετικές ταχύτητες επεξεργαστή.

Αντιμετώπιση του κατακερματισμού μέσω της συνεχής βελτίωσης του προγράμματος της JCP

Το πρόγραμμα της JCP είναι μια ευέλικτη διαδικασία που αφήνει πολλές λεπτομέρειες στην διαδικασία της προδιαγραφής μέχρι την διάκριση του από τον Specification Lead. Για παράδειγμα, οι διαδικασίες λήψης αποφάσεων μπορούν να ποικίλλουν από το ένα JSR στο άλλο, σημαντικά. Το ύφος και η μορφή που χρησιμοποιείται για την προδιαγραφή την ίδια μπορεί να διαφέρει αρκετά. Ομοίως, οι

απαιτήσεις δοκιμών για την κάλυψη TCK δεν ορίζονται με μεγάλη σαφήνεια, και, ως εκ τούτου, η δοκιμές των TCKs σε διάφορετικά πρότυπα του JCP προγράμματος μπορεί να ποικίλλει σημαντικά.

[1]

7.3 Ανακεφαλαίωση

Σε αυτό το κεφάλαιο είδαμε το φαινόμενο του Κατακερματισμού, από ποια είδη αποτελείται και πως μπορεί να επιλυθεί. Στο επόμενο κεφάλαιο, ο αναγνώστης θα μάθει για τα vCards, τα iCalendars κι άλλα είδη format για την αποθήκευση δεδομένων.

Κεφάλαιο 8ο

8.1 Εισαγωγή

Σε αυτό το κεφάλαιο θα δούμε τι είναι τα formats vCard, iCalendar κι ορισμένα άλλα, τα οποία αποθηκεύουν PIM πληροφορίες. Το πρώτο αφορά τις επαφές, ενώ το δεύτερο αφορά αρχεία που έχουν σχέση με το ημερολόγιο. Ακόμη γίνεται κατανοητή και η δομή των παραπάνω αρχείων, κάτι πολύ σημαντικό σε περίπτωση που κάποιος κοιτάξει τον κώδικα της εργασίας αυτής.

8.2 vCard

8.2.1 Εισαγωγή στο vCard

Το vCard είναι μία μορφή αρχείου, πρότυπη για ηλεκτονικές business cards. Τα vCards είναι συχνά επισυννημένα σε e-mail μηνύματα, αλλά μπορούν να σταλθούν με διάφορους τρόπους, όπως μέσω του Παγκοσμίου Ιστού. Ένα vCard μπορεί να περιέχει πληροφορίες, όπως όνομα, διεύθυνση, αριθμούς τηλεφώνων, urls, σήματα, φωτογραφίες, ακόμα και ήχους.

Το vCard ή Versitcard, προτάθηκε το 1995 από το Versit cosortium, το οποίο αποτελούντο από τις εταιρείες Apple, AT&T Technologies (αργότερα Lucent), IBM και Siemens. Τον Δεκέμβριο του 1996, η ιδιοκτησία της μορφής αυτής, δόθηκε στο Internet Mail Consortium, μία εμπορική ένωση από εταιρείες με ενδιαφέρον στο e-mail.

Το vCard συνοδεύεται από ένα προτεινόμενο πρότυπο για ανταλλαγή δεδομένων για προσεχείς συναντήσεις, οι οποίες ονομάζονται vCalendar, δεδομένου ότι αντικαταστάθηκαν από το iCalendar. Το Internet Mail Consortium δήλωσε πως ελπίζει ότι όλοι όσοι αναπτύσσουν vCards θα υιοθετήσουν το πρότυπό τους και θα είναι συμβατά με το vCalendar 1.0 και το iCalendar.

Η έκδοση 2.1 του προτύπου vCard υποστηρίζεται ευρέως από τους e-mail clients. Η έκδοση 3.0 της μορφής αυτής είναι είναι μία πρόταση IETF standards-track που περιλαμβάνεται στα RFC 2425 και RFC 2426. Η πιο συνηθισμένη προέκταση αρχείου της μορφής vCard είναι το .vcf

Διάφορες εφαρμογές έχουν διαφορετικές υλοποιήσεις του προτύπου vCard. Στο Mac OS X στην εφαρμογή Address Book επιτρέπεται εξαγωγή όλων των επαφών σε ένα vcf αρχείο, ενώ στο Microsoft Outlook επιτρέπεται μόνο μία επαφή σε κάθε αρχείο. Στις διανομές Linux, υπάρχουν διάφορα προγράμματα, όπως το Kontact και το Evolution τα οποία επιτρέπουν κατά την εξαγωγή μίας ή περισσοτέρων επαφών ανά αρχείο.

8.2.2 Παραδείγματα vCard αρχείων

vCard 2.1

```
BEGIN:VCARD
VERSION:2.1
N:Gump;Forrest
FN:Forrest Gump
ORG:Bubba Gump Shrimp Co.
TITLE:Shrimp Man
TEL;WORK;VOICE:(111) 555-1212
TEL;HOME;VOICE:(404) 555-1212
ADR;WORK:;;100 Waters Edge;Baytown;LA;30314;United States of America
LABEL;WORK;ENCODING=QUOTED-PRINTABLE:100 Waters Edge=0D=0ABaytown, LA
30314=0D=0AUnited States of America
ADR;HOME:;;42 Plantation St.;Baytown;LA;30314;United States of America
LABEL;HOME;ENCODING=QUOTED-PRINTABLE:42 Plantation St.=0D=0ABaytown, LA
30314=0D=0AUnited States of America
EMAIL;PREF;INTERNET:forrestgump@example.com
REV:20080424T195243Z
END:VCARD
```

vCard 3.0

```
BEGIN:VCARD
VERSION:3.0
N:Gump;Forrest
FN:Forrest Gump
ORG:Bubba Gump Shrimp Co.
TITLE:Shrimp Man
TEL;TYPE=WORK,VOICE:(111) 555-1212
TEL;TYPE=HOME,VOICE:(404) 555-1212
ADR;TYPE=WORK:;;100 Waters Edge;Baytown;LA;30314;United States of America
LABEL;TYPE=WORK:100 Waters Edge\nBaytown, LA 30314\nUnited States of America
ADR;TYPE=HOME:;;42 Plantation St.;Baytown;LA;30314;United States of America
LABEL;TYPE=HOME:42 Plantation St.\nBaytown, LA 30314\nUnited States of America
EMAIL;TYPE=PREF,INTERNET:forrestgump@example.com
REV:20080424T195243Z
END:VCARD
```

Το vCard ορίζει τις παρακάτω ιδιότητες τύπων: FN, N, NICKNAME, PHOTO, BDAY, ADR, LABEL, TEL, EMAIL, MAILER, TZ, GEO, TITLE, ROLE, LOGO, AGENT, ORG, CATEGORIES, NOTE, PRODID, REV, SORT-STRING, SOUND, URL, UID, VERSION, CLASS, KEY

Πίνακας 8.1 - Ιδιότητες τύπων του vCard format

Όνομα	Περιγραφή	Σημασία
N	Όνομα	μία δομημένη αναπαράσταση του ονόματος του ατόμου, μέρους ή πράγματα το οποίο σχετίζεται με το vCard αντικείμενο
FN	Formatted όνομα	το όνομα το οποίο σχετίζεται με το vCard αντικείμενο σε formatted string
PHOTO	Φωτογραφία	μία εικόνα ή φωτογραφία του ατόμου το οποίο σχετίζεται με το vCard
BDAY	Γενέθλεια	η ημερομηνία γέννησης του ατόμου το οποίο σχετίζεται με το vCard
ADR	Διεύθυνση	μία δομημένη αναπαράσταση της φυσικής διεύθυνσης το ατόμου ή του αντικειμένου με το οποίο σχετίζεται το vCard
LABEL	Ετικέτα διεύθυνσης	μία ετικέτα για την διεύθυνση του ατόμου με το οποίο σχετίζεται το vCard
TEL	Τηλέφωνο	ο αριθμός τηλεφώνου του για τηλεφωνική επικοινωνία με το αντικείμενο του vCard αντικειμένου
EMAIL	Email	η ηλεκτρονική διεύθυνση για επικοινωνία με το vCard αντικείμενο
MAILER	Email πρόγραμμα (προαιρετικό)	ο τύπος του email προγράμματος που χρησιμοποιείται
TZ	Time Zone	οι πληροφορίες οι οποίες σχετίζονται με το standard time zone του vCard του αντικειμένου
GEO	Παγκόσμια θέση	η ιδιότητα η οποία καθορίζει ένα γεωγραφικό μήκος και πλάτος
TITLE	Τίτλος	διευκρινίζει τον τίτλο επαγγέλματος, την θέση του ή την λειτουργία με την οποίοα σχετίζεται το vCard αντικείμενο μέσα σε έναν οργανισμό / επιχείρηση
ROLE	Ρόλος ή απασχόληση	ο ρόλος, η απασχόληση ή κατηγορία επαγγέλματος, του vCard αντικειμένου μέσα σε έναν οργανισμό / επιχείρηση
LOGO	Λογότυπο	μία εικόνα ή ένα γραφικό του λογοτύπου το οργανισμού / επιχείρηση με την οποία συσχετίζεται το αντικείμενο στο οποίο ανήκει το vCard
AGENT	Agent	πληροφορίες για ένα άλλο άτομο το οποίο θα δράσει εκ μέρους του vCard αντικειμένου
ORG	Όνομα οργανισμού ή μονάδα οργανισμού	το όνομα και προαιρετικά οι μονάδες του οργανισμού με τις οποίες σχετίζεται το vCard αντικείμενο. Αυτή η ιδιότητα βασίζεται στο X.520 Organization Name attribute και στο X.520 Organization Unit attribute
NOTE	Σημείωση	διευκρινίζειι επιπρόσθετες πληροφορίες ή ένα σχόλιο το οποίο σχετίζεται με το vCard
REV	Τελευταία	συνδυασμός της ημερομηνίας του ημερολογίου και της ώρας για

Όνομα	Περιγραφή	Σημασία
	αναθεώρηση	την τελευταία ανανέωση του vCard αντικειμένου
SOUND	Ήχος	από προεπιλογή, εάν αυτή η ιδιότητα δεν είναι ομαδοποιημένη με άλλες ιδιότητες ορίζει την προφορά της Formatted Name ιδιότητας του vCard αντικειμένου
URL	URL	ένα URL είναι η αναπαράσταση ενός διαδικτυακού τόπου ο οποίος μπορεί να περιέχει πληροφορίες πραγματικού χρόνου για το vCard αντικείμενο
UID	Μοναδικό αναγνωριστικό	ορίζει μία τιμή η οποία αναπαριστά ένα παγκοσμίως μοναδικό αναγνωριστικό για το αντικείμενο
VERSION	Έκδοση	έκδοση της Vcard προδιαγραφής
KEY	Δημόσιο κλειδί	το δημόσιο κλειδί κρυπτογράφισης το οποίο σχετίζεται με το vCard αντικείμενο

[54]

8.3 iCalendar

8.3.1 Εισαγωγή στο iCalendar

Το iCalendar είναι μία μορφή αρχείου, η οποία επιτρέπει στους χρήστες του Internet να στέλνουν αιτήσεις συνάντησης και διεργασία προς άλλους χρήστες του Internet, μέσω e-mail, ή διαμοιραζόμενοι αρχεία με την κατάληξη .ics . Οι παραλήπτες δεδομένων iCalendar μπορούν να απαντήσουν στον αποστολέα εύκολα με χρήση συμβατών προγραμμάτων.

Το iCalendar χρησιμοποιείται και υποστηρίζεται από έναν μεγάλο αριθμό προϊόντων. Τα αρχεία αυτά μπορούν να σταλούν με διάφορους τρόπους και πιο συχνά αυτό γίνεται με την χρήση e-mail.

8.3.2 Ιστορία και σχεδιασμός

Το iCalendar δημιουργήθηκε από το Internet Engineering Task Force Calendaring and Scheduling Working Group και συνεγράφει από τον Frank Dawson της Lotus Development Corporation και τον Derik Stenerson της Microsoft Corporation. Το iCalendar βασίζεται κυρίως στο πρώιμο vCalendar, το οποίο δημιουργήθηκε από το Internet Mail Consortium. Τα αρχεία δεδομένων του iCalendar είναι plain text.

8.3.3 Περιορισμοί και το μέλλον

Η μορφή iCalendar σχεδιάστηκε να στέλνει δεδομένα βασισμένα στο ημερολόγιο, όπως events κι εσκεμμένα δεν περιγράφει τι πρέπει να γίνει με αυτά τα δεδομένα.

Το iCalendar έχει ως στόχο την παροχή του ορισμού μίας κοινής μορφή για ανοιχτή αναταλλαγή πληροφοριών ημερολογίου και χρονοπρογραμματισμού σε μέσα το Internet. Παρά το γεγονός ότι τα πιο πολλά χαρακτηριστικά τα οποία χρησιμοποιούνται ευρέως από τους χρήστες υποστηρίζονται από το iCalendar, μερικές πιο προηγμένες ικανότητες έχουν πρόβλημα. Για παράδειγμα, οι πιο πολλοί κατασκευαστές δεν υποστηρίζουν Journals (VJOURNAL). Ακόμη, η επανάληψη συναντήσεων ως προς τον κανόνα της επανάληψης δημιουργεί αρκετά προβλήματα σε όσους ασχολούνται με αυτές, κυρίως με τις μετατροπές τους. Τα VTODOS έχουν επίσης πρόβλημα.

Επίσης, το ημερολόγιο του iCalendar δεν είναι συμβατό με μερικά μη Γρηγοριανά ημερολόγια, όπως τα σεληνιακά ημερολόγια.

Το memo “Calendar Access Protocol” (RFC 4324) ήταν μία αρχική προσπάθεια σε ένα παγκόσμιο σύστημα για την δημιουργία ημερολογία πραγματικού χρόνου. Το πρωτόκολο αυτό τελικά εγκαταλήφθηκε εξαιτίας της τεράστιας πολυπλκότητάς του. Παρ'όλα αυτά, κώδικας όπως το GroupDav και CalDav, βασισμένος στο iCalenar χρησιμοποιείται πιο συχνά σε πακέτα λογισμικού σε server και client.

Το IETF Calendaring and Scheduling (calsify) Working Group ήταν στην διαδικασία αναθεώρησης των προτύπων iCalendar.

8.3.4 Τεχνικές προδιαγραφές

Η πιο υψηλού επιπέδου γλώσσα προγραμματισμού για το iCalendar είναι η Calendaring and Scheduling Core Object. Μία συλλογή από πληροφορίες ημερολογίου και χρονοπρογραμματισμού. Τυπικά οι πληροφορίες θα αποτελούνται από ένα μοναδικό iCalendar object. Πάρα ταύτα όμως, πολλαπλά iCalendar αντικείμενα, μπορούν να ομαδοποιηθούν μαζί.

Η πρώτη γραμμή θα πρέπει να ξεκινάει “BEGIN:VCALENDAR” και η τελευταία γραμμή θα πρέπει να είναι “END:VACALENDAR”. Το σώμα του κώδικα μεταξύ των δύο αυτών γραμμών ονομάζεται icalbody.

Το icalbody αποτελείται από μία λίστα ιδιοτήτων ημερολογίου, και ένα ή περισσότερα συνιστώντα μέρη ημερολογίου. Οι ιδιότητες ημερολογίους εφαρμόζονται σε ολόκληρο το ημερολόγιο. Τα συνιστώντα μέρη είναι μερικές ιδιότητες ημερολογίου τα οποία δημιουργούν ένα “σχέδιο” ημερολογίου. Για παράδειγμα, ένα συνιστόν μέρος μπορεί να προσδιορίζει ένα event, μία to-do list, ένα journal entry, time zone information, or free/busy time information, or an alarm.

Παρακάτω παρατείθεται ένα απλό παράδειγμα ενός iCalendar αντικειμένου με θέμα το Bastille Day Party, το οποίο έλαβε χώρα στις 14 Ιουλίου 1997 στις 17:00:00 και τελείωσε στις 15 Ιουλίου 1997 στις 03:59:59.

```
BEGIN:VCALENDAR
VERSION:2.0
PRODID:-//hacksw/handcal//NONSGML v1.0//EN
BEGIN:VEVENT
DTSTART:19970714T170000Z
DTEND:19970715T035959Z
SUMMARY:Bastille Day Party
END:VEVENT
END:VCALENDAR
```

Γενικά υπάρχουν πολλά είδη component τα οποία μπορούν να χρησιμοποιηθούν στο iCalendar.

Events (VEVENT)

Το "VEVENT" περιγράφει ένα event, το οποίο έχει προγραμματισθεί κάποια χρονική στιγμή σε ένα ημερολόγιο. Ένα VEVENT μπορεί να περιέχει ένα VALARM το οποίο να επιτρέπει την χρήση ενός alarm. Τέτοια events έχουν ένα DTSTART το οποίο θέτει τον χρόνο εκκίνησης του event και ένα DTEND το οποίο θέτει τον χρόνο τέλους του. Εάν το event έχει κανόνας επανάληψης, το DTSTART θέτει την έναρξη της πρώτης εμφάνισης του event.

Το VEVENT επίσης χρησιμοποιείται για events ημερολογίου χωρίς μία συγκεκριμένη χρονική στιγμή, όπως επετείους και καθημερινές υπενθυμίσεις. Εάν πρέπει να ακυρωθεί ένα event, τότε το UID του, θα πρέπει να είναι ίδιο με εκείνο του αρχικού event και οι ιδιότητες του component θα πρέπει να τεθούν στην ακύρωση.

METHOD:CANCEL
STATUS:CANCELLED

Για να στείλεις κάποιος ένα UPDATE για ένα event τότε θα πρέπει το UID του να είναι ίδιο με το UID του αρχικού event.

SEQUENCE:<Num of Update>

π.χ. για το πρώτο update
SEQUENCE:1

Στο Outlook της Microsoft, το πεδίο SUMMARY αναπαριστά το θέμα για κάθε είσοδο στην φόρμα "Appointment" και το πεδίο DESCRIPTION στο κείμενο περιγραφής του γεγονότος. Επί προσθέτως, το Outlook 2003 απαιτεί ένα UID κι ένα DTSTAMP.

To-do (VTODO)

Το VTODO επεξηγεί ένα στοιχείο to-do, όπως μία πράξη, ή μία υποχρέωση.

Το παρακάτω παράδειγμα είναι ένα to-do το οποίο πρέπει να γίνει μέχρι τις 15 Απριλίου 1998 στις

20:00:00. Ένα alarm με ήχο θα χτυπήσει την προηγούμενη μέρα, στις 12 το μεσημέρι και για 4 φορές, κάθε 1 ώρα για να το υπενθυμίσει στον χρήστη.

```
BEGIN:VCALENDAR
VERSION:2.0
PRODID:-//ABC Corporation//NONSGML My Product//EN
BEGIN:VTODO
DTSTAMP:19980130T134500Z
SEQUENCE:2
UID:uid4@host1.com
ORGANIZER:MAILTO:unclesam@us.gov
ATTENDEE;PARTSTAT=ACCEPTED:MAILTO:jqpublic@example.com
DUE:19980415T235959
STATUS:NEEDS-ACTION
SUMMARY:Submit Income Taxes
BEGIN:VALARM
ACTION:AUDIO
TRIGGER:19980414T120000
ATTACH;FMTTYPE=audio/basic:http://example.com/pub/audio-files/ssbanner.aud
REPEAT:4
DURATION:PT1H
END:VALARM
END:VTODO
END:VCALENDAR
```

Journal entry (VJOURNAL)

Το VJOURNAL είναι ένα journal entry. Προστείθεται ένα περιγραφικό κείμενο σε μία συγκεκριμένη ημερομηνία και μπορεί να χρησιμοποιηθεί για να καταγράψει ένα καθημερινό σύνολο από δραστηριότητες ή να περιγράψεις μία διαδιακσία με μία εγγραφή σχετική με το to-do.

Το παρακάτω είναι ένα παράδειγμα ενός journal entry:

```
BEGIN:VCALENDAR
VERSION:2.0
PRODID:-//ABC Corporation//NONSGML My Product//EN
BEGIN:VJOURNAL
DTSTAMP:19970324T120000Z
UID:uid5@host1.com
ORGANIZER:MAILTO:jsmith@example.com
STATUS:DRAFT
CLASS:PUBLIC
CATEGORY:Project Report, XYZ, Weekly Meeting
DESCRIPTION:Project xyz Review Meeting Minutes\n
 Agenda\n1. Review of project version 1.0 requirements.\n2.
 Definition
 of project processes.\n3. Review of project schedule.\n
 Participants: John Smith, Jane Doe, Jim Dandy\n-It was
  decided that the requirements need to be signed off by
```

```
  product marketing.\n-Project processes were accepted.\n
 -Project schedule needs to account for scheduled holidays
  and employee vacation time. Check with HR for specific
  dates.\n-New schedule will be distributed by Friday.\n-
 Next weeks meeting is cancelled. No meeting until 3/23.
END:VJOURNAL
END:VCALENDAR
```

Free/busy time (VFREEBUSY)

VFREEBUSY είναι μία αίτηση για free/busy time, είναι μία απάντηση σε αίτηση ή είναι ένα σύνολο από busy time

Το παρακάτω είναι ένα παράδειγμα για busy time πληροφορίες:

```
BEGIN:VCALENDAR
VERSION:2.0
PRODID:-//RDU Software//NONSGML HandCal//EN
BEGIN:VFREEBUSY
ORGANIZER:MAILTO:jsmith@example.com
DTSTART:19980313T141711Z
DTEND:19980410T141711Z
FREEBUSY:19980314T233000Z/19980315T003000Z
FREEBUSY:19980316T153000Z/19980316T163000Z
FREEBUSY:19980318T030000Z/19980318T040000Z
URL:http://www.host.com/calendar/busytime/jsmith.ifb
END:VFREEBUSY
END:VCALENDAR
```

vCalendar 1.0

Το σχέδιο του iCalendar βασίστηκε στο προηγούμενο format τύπου vCalendar το οποίο δημιουργήθηκε από το Internate Mail Consortium

Το παρακάτω είναι ένα παράδειγμα μορφή vCalendar:

```
BEGIN:VCALENDAR
VERSION:1.0
BEGIN:VEVENT
CATEGORIES:MEETING
STATUS:TENTATIVE
DTSTART:19960401T033000Z
DTEND:19960401T043000Z
SUMMARY:Your Proposal Review
DESCRIPTION:Steve and John to review newest proposal material
CLASS:PRIVATE
END:VEVENT
END:VCALENDAR
```

Άλλοι τύποι components

Άλλοι τύποι component περιλαμβάνουν VTIMEZONE (time zones) και VALARM (alarms). Κάποια components μπορεί να περιλαμβάνουν άλλα components. Π.χ. το VALARM περιλαμβάνεται συχνά σε άλλα components.

Ανανεώσεις

Το πεδίο UID κάνει updates όταν τα χρονοπρογραμματισμένα event αλλάζουν. Όταν παράγεται για πρώτη φορά ένα event ένα παγκοσμίως μοναδικό αναγνωριστικό δημιουργείται. Εάν πρέπει να ξαναγίνει ένα event με το ίδιο UID, τότε θα αντικαταστήσει το αρχικό.

[43]

8.4 Ανακεφαλαίωση

Σε αυτό το κεφάλαιο είδαμε τα formats τα οποία αφορούν αποθήκευση πληροφοριών από κινητές συσκευές κι όχι μόνο. Στο παρακάτω κεφάλαιο γίνεται η UML παρουσίαση του προγράμματος.

Κεφάλαιο 9ο

9.1 Εισαγωγή

Σε αυτό το κεφάλαιο θα γίνει μία μικρή ανάλυση του προγράμματος με χρήστη της UML. Ο αναγνώστης θα δει την οργάνωση της εφαρμογής με την βοήθεια του class diagram, του activity diagram, του use case diagram, του sequence diagram, καθώς με με μία πλήρη παρουσίαση κάθε class με μία μικρή περιγραφή της κάθε μεθόδου που την αποτελεί.

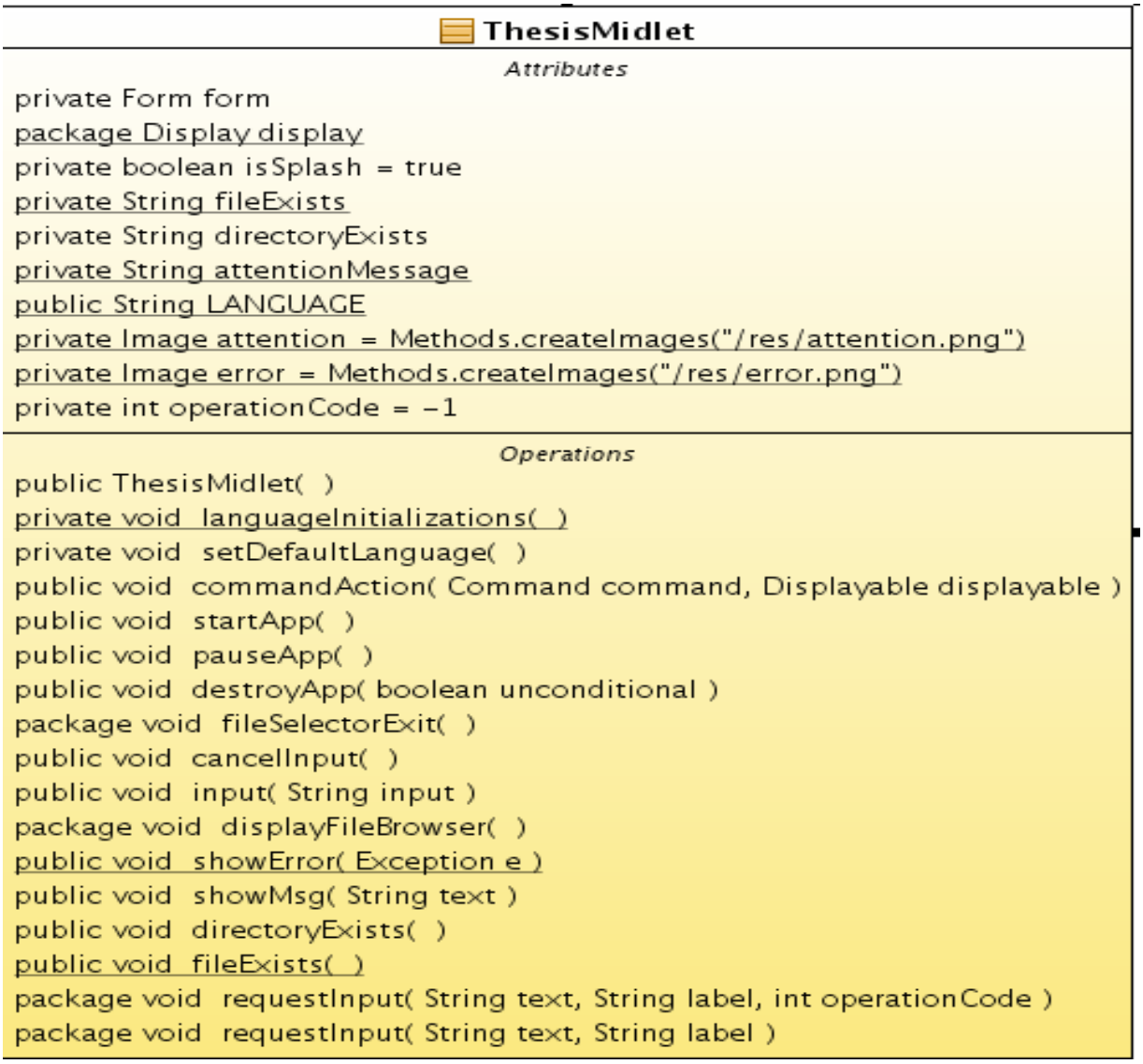
Για την δημιουργία των παρακάτω διαγραμμάτων, χρησιμοποιήθηκε το plug-in το οποίο υπάρχει για τα Netbeans 6.5 και Netbeans 6.7, μέσω της δυνατότητας του reverse engineering που έχει.

Λόγω της χρήσης του παραπάνω εργαλείου, υπάρχουν κάποιες διαφορές στην παρουσίαση των διαφόρων συστατικών ενός class. Οτιδήποτε δηλώνεται ως static, αναπαρίσταται στα διαγράμματα υπογραμμισμένο. Ενώ οι αναφορές ως προς το visibility, γίνεται με τις χρήσεις λέξεων κι όχι συμβόλων. Δηλαδή μία μεταβλητή η οποία δηλώνεται ως public, θα γράφει μπροστά “public” και δεν θα κάνει την χρήση του συμβόλου “+”.

9.2 UML

9.2.1 Package backbone

Class ThesisMidlet



Εικόνα 9.1 - class ThesisMidlet

Η class ThesisMidlet είναι το κύριο και μοναδικό MIDlet του προγράμματος. Από εδώ γίνεται η έναρξη και ο τερματισμός του. Εδώ υπάρχουν μερικές από τις πιο σημαντικές μεθόδους, ενώ οι περισσότερες classes συνδέονται με αυτήν. Η class ThesisMidlet κάνει extend το MIDlet και implement το CommandListener. Ανήκει στο package backbone, μίας και είναι η πιο σημαντική class κι αποτελεί το πρώτο μέρος της ραχοκοκαλιάς (backbone) του προγράμματος.

Ο constructor δεν έχει κανένα όρισμα, κάνει αρχικοποίηση της form πάνω στην οποία βασίζεται το πρόγραμμα και του FileSelector μέσω του οποίο γίνεται το browsing, ενώ με την κλήση της μεθόδου *private void languageInitializations()* φορτώνονται οι default ρυθμίσεις για την γλώσσα.

Η μέθοδος *public void fileSelectorExit()* τερματίζει την λειτουργία του FileSelector.

Η μέθοδος *public void cancelInput()* ακυρώνει την διαδικασία εισαγωγής κάποιου ονόματος, για την δημιουργία ενός φακέλου ή αρχείου, από το χρήστη.

Η μέθοδος *public void input(String input)* παίρνει το όνομα που δίνει ο χρήστης για την δημιουργία ενός φακέλου ή αρχείου.

Η μέθοδος *publicvoid displayFileBrowser()* παρουσιάζει τα περιεχόμενα του file system της κινητής συσκευής.

Η μέθοδος *public static void showError(Exception e)* εμφανίζει μήνυμα σφάλματος στην οθόνη.

Η μέθοδος *public void showMsg(String text)* δείχνει στο χρήστη ένα μήνυμα για την πληροφόρισή του πάνω σε μία πιθανώς εσφαλμένη ή μη επιτρεπτή ενέργεια.

Η μέθοδος *public void directoryExists()* δείχνει στον χρήστη ένα μήνυμα, ενημερώνοντάς τον πως το όνομα του φακέλου που έδωσε για την δημιουργία του είναι ήδη ενός υπάρχοντα φακέλου.

Η μέθοδος *public void fileExists()* δείχνει στον χρήστη ένα μήνυμα, ενημερώνοντάς τον πως το όνομα του φακέλου που έδωσε για την δημιουργία του είναι ήδη ενός υπάρχοντα φακέλου.

Η μέθοδος *public void requestInput(String text, String label, int operationCode)* ζητά από τον χρήστη να εισάγει το όνομα ενός φακέλου ή αρχείου.

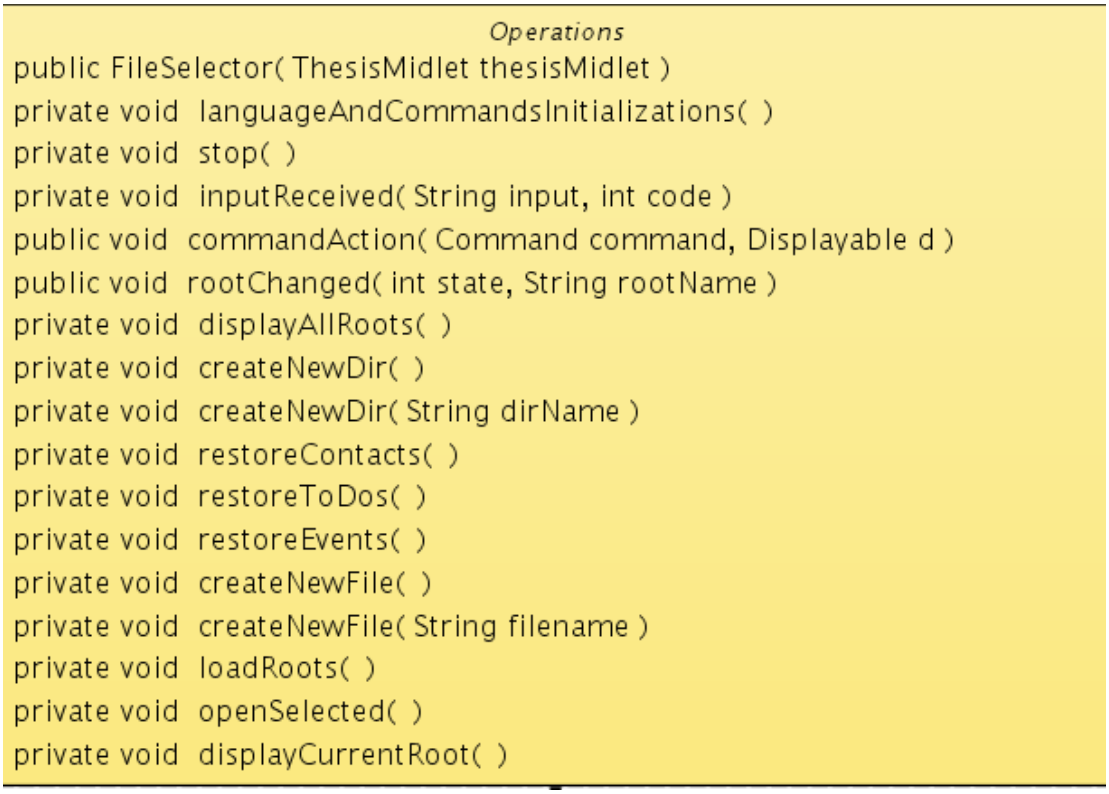
Η μέθοδος *public void requestInput(String text, String label)* ζητά από τον χρήστη να εισάγει το όνομα ενός φακέλου ή αρχείου.

Class FileSelector

FileSelector

Attributes

private String open
private String back
private String exit
private String createFile
private String createDirectory
private String restoreToDos
private String restoreContacts
private String restoreEvents
private String cannotCreateFile
private String cannotCreateDirectory
private Image ROOT_IMAGE = Methods.createImages("/res/root.png")
private Image FOLDER_IMAGE = Methods.createImages("/res/folder.png")
private Image FILE_IMAGE = Methods.createImages("/res/file.png")
private String FILE_SEPARATOR = System.getProperty("file.separator") != null
private String UPPER_DIR = ".."
private String FILE = "file:///"
private Command openCommand
private Command createDirCommand
private Command exitCommand
private Command createFileCommand
private Command backCommand
private Command restoreContactsCommand
private Command restoreEventsCommand
private Command restoreToDosCommand
private int MKDIR_OP = 1
private int INIT_OP = 2
private int OPEN_OP = 3
private int GIVE_NAME = 5
private int BACK = 4
private int REST_TODO = 8
private int REST_CONT = 6
private int REST_EVENT = 7
private Vector rootsList = new Vector()
private FileConnection currentRoot = null
private String path
private String newFileName
private String newDirectoryName
private ThesisMidlet thesismidlet



Εικόνα 9.2 - class FileSelector

Η class FileSelector έχει πρόσβαση στο file system της κινητής συσκευής κι έτσι δείχνει μέσα στο πρόγραμμα την ιεραρχία των καταλόγων και των αρχείων. Η class αυτή κάνει extend την List και implement το CommandListener καθώς και το FileSystemListener. Ανήκει στο package αυτό, μιας κι αποτελεί το δεύτερο μέρος της ραχοκοκαλιάς (backbone) του προγράμματος.

Ο constructor έχει σαν όρισμα το MIDlet, την ThesisMidlet, ενώ με την κλήση της μεθόδου *private void languageAndCommandsInitializations()* μπορεί και φορτώνει τις επιλογές για την γλώσσα που έχει επιλεχθεί.

Η μέθοδος *private void stop()* στην ουσία καλεί την μέθοδο abort() της class OperationsQueue, προκειμένου να σταματήσει το FileSelector αντικείμενο.

Η μέθοδος *private void inputReceived(String input, int code)* ζητάει από τον χρήστη την εισαγωγή κάποιου ονόματος.

Η μέθοδος *public void rootChanged(int state, String rootName)* αντιλαμβάνεται την όποια αλλαγή μέσα στο root του file system.

Η μέθοδος *private void displayAllRoots()* απεικονίζει μέσα στο πρόγραμμα την ιεραρχία καταλόγων

που υπάρχει στο file system της κινητής συσκευής.

Η μέθοδος *private void createNewDir()* δημειουργεί ένα φάκελο.

Η μέθοδος *private void createNewDir(String dirName)* δημιουργεί ένα φάκελο με όνομα δοσμένο από τον χρήστη.

Η μέθοδος *private void restoreContacts()* είναι εκείνη που κάνει το restore των contacts.

Η μέθοδος *private void restoreEvents()* είναι εκείνη που κάνει το restore των events.

Η μέθοδος *private void restoreToDos()* είναι εκείνη που κάνει το restore των todos

Η μέθοδος *private void createNewFile()* δημειουργεί ένα αρχείο.

Η μέθοδος *private void createNewFile(String filename)* δημιουργεί ένα αρχείο με όνομα δοσμένο από τον χρήστη.

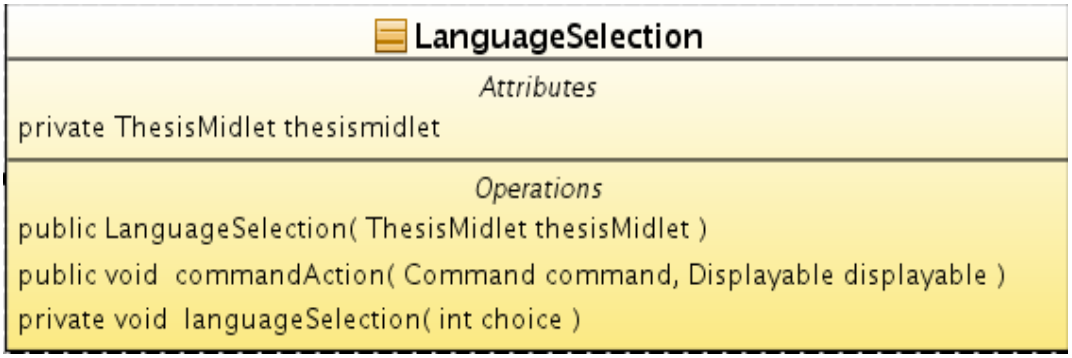
Η μέθοδος *private void loadRoots()* φορτώνει όλα τα περιεχόμενα της ιεραρχίας καταλόγων του file system της κινητής συσκευής.

Η μέθοδος *private void openSelected()* ανοίγει τους καταλόγους της κινητής συσκευής.

Η μέθοδος *private void displayCurrentRoot()* απεικονίζει τα περιεχόμενα του τρέχοντος καταλόγου στον οποίο βρίσκεται ο χρήστης.

9.2.2 Package selections

Class LanguageSelection



Εικόνα 9.3 - class LanguageSelection

Η class LanguageSelection είναι εκείνη μέσω της οποίας επιλέγουμε το τι θα κάνουμε μέσα στο

πρόγραμμα. Αυτή η class κάνει extend την List και κάνει implement το CommandListener. Ανήκει στο package selections, καθώς ο χρήστης επιλέγει τα στοιχεία της, τα οποία εμφανίζονται στην οθόνη.

Ο constructor έχει σαν όρισμα το MIDlet, την ThesisMidlet, ενώ με την κλήση της μεθόδου *private void languageAndCommandsInitializations()* μπορεί και φορτώνει τις επιλογές για την γλώσσα που έχει επιλεχθεί. Κάθε στοιχείο της λίστας έχει και φορτωμένη μία εικόνα, δείχνοντας έτσι καλύτερα τις δυνατότητες επιλογής της κάθε γλώσσας.

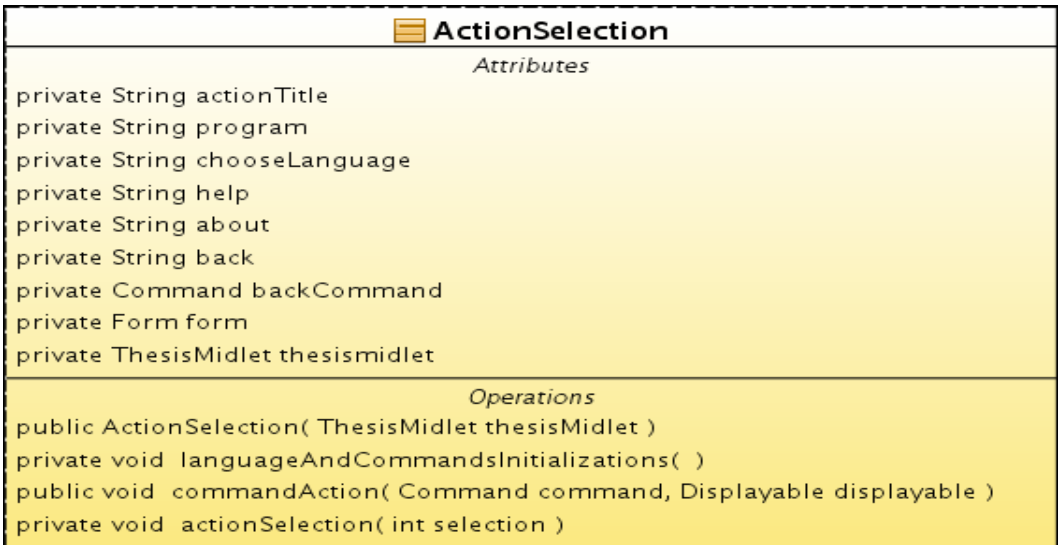
Οι επιλογές των γλωσσών τις οποίες μας δίνει αυτή η class, είναι:

- Ελληνικά
- Αγγλικά
- Ισπανικά
- Γαλλικά
- Γερμανικά
- Ιταλικά
- Έξοδος από το πρόγραμμα

Οποιαδήποτε γλώσσα κι αν επιλεχθεί, τότε φορτώνεται στην συνέχεια η class ActionSelection, ενώ με την επιλογή εξόδου από το πρόγραμμα, καλείται η μέθοδος *public void destroyApp(boolean flag)* του MIDlet.

Ο έλεγχος όλων των παραπάνω επιλογών γίνεται με την χρήση της μεθόδου *private void languageSelection(int selection)* η οποία χρησιμοποιώντας την δομή επιλογής switch, καλεί την setLanguage() μέθοδο για να φορτωθεί το αρχείο της επιθυμητής γλώσσας.

Class ActionSelection



Εικόνα 9.4 - class ActionSelection

Η class ActionSelection είναι εκείνη μέσω της οποίας επιλέγουμε το τι θα κάνουμε μέσα στο πρόγραμμα. Αυτή η class κάνει extend την List και κάνει implement το CommandListener. Ανήκει στο package selections, καθώς ο χρήστης επιλέγει τα στοιχεία της, τα οποία εμφανίζονται στην οθόνη.

Ο constructor έχει σαν όρισμα το MIDlet, την ThesisMidlet, ενώ με την κλήση της μεθόδου *private void languageAndCommandsInitializations()* μπορεί και φορτώνει τις επιλογές για την γλώσσα που έχει επιλεχθεί.

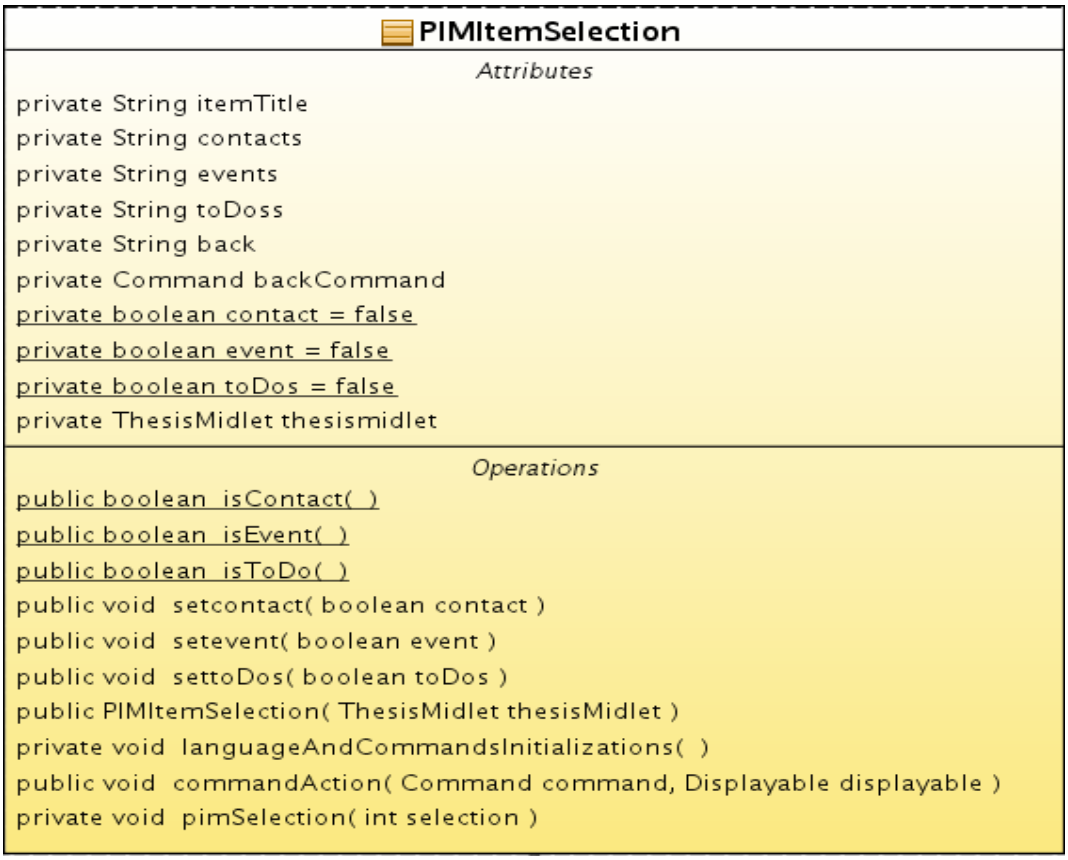
Οι επιλογές τις οποίες μας δίνει αυτή η class, είναι:

- Να προχωρήσουμε στην χρήση του προγράμματος για να κάνουμε backup ή restore των δεδομένων
- Να αλλάξουμε την γλώσσα του προγράμματος
- Να δούμε το κείμενο βοήθειας για την χρήση του προγράμματος
- Να δούμε πληροφορίες σχετικά με το πρόγραμμα
- Να επιλέξουμε τον τερματισμό του προγράμματος

Σε περίπτωση που επιλέξουμε την συνέχιση του προγράμματος θα κληθεί η class PIMItemSelection. Ενώ στην επιλογή γλώσσας θα ξαναφορτωθεί η class LanguageSelection. Στο κείμενο όμως βοηθείας και πληροφοριών φορτώνεται μία Form, η οποία θα μας απεικονίσει όλα τα περιεχόμενα σχετικά με αυτούς του τομείς, ανάλογα με την γλώσσα την οποία είχαμε επιλέξει προηγουμένως. Επιλέγοντας τον τερματισμό του προγράμματος θα κληθεί η μέθοδος *public void destroyApp(boolean flag)* του MIDlet.

Ο έλεγχος όλων των παραπάνω επιλογών γίνεται με την χρήση της μεθόδου *private void actionSelection(int selection)* η οποία χρησιμοποιώντας την δομή επιλογής switch, μας δίνει το αποτέλεσματα μία ενέργειας, για την κάθε επιλογή.

Class PIMItemSelection



Εικόνα 9.5 - class PIMItemSelection

Η class PIMItemSelection είναι εκείνη μέσω της οποίας επιλέγουμε το πιο από τα μέρη του PIM (Contacts, Events, ToDos) θα κάνουμε backup ή restore. Αυτή η class κάνει extend την List και κάνει implement το CommandListener. Ανήκει στο package selections, καθώς ο χρήστης επιλέγει τα στοιχεία της, τα οποία εμφανίζονται στην οθόνη.

Ο constructor έχει σαν όρισμα το MIDlet, την ThesisMidlet, ενώ με την κλήση της μεθόδου *private void languageAndCommandsInitializations()* μπορεί και φορτώνει τις επιλογές για την γλώσσα που έχει επιλεχθεί.

Υπάρχουν οι μέθοδοι *public void setContact(boolean contact), public void setEvent(boolean event)* και *public void setEvent(boolean restore)* ώστε να κάνουμε set το αν έχει γίνει επιλογή για contact, event ή todo ενώ με τις μεθόδους *public static boolean isContact(), public static boolean isEvent()* και *public static boolean isToDo()* βλέπουμε την επιλογή του χρήστη.

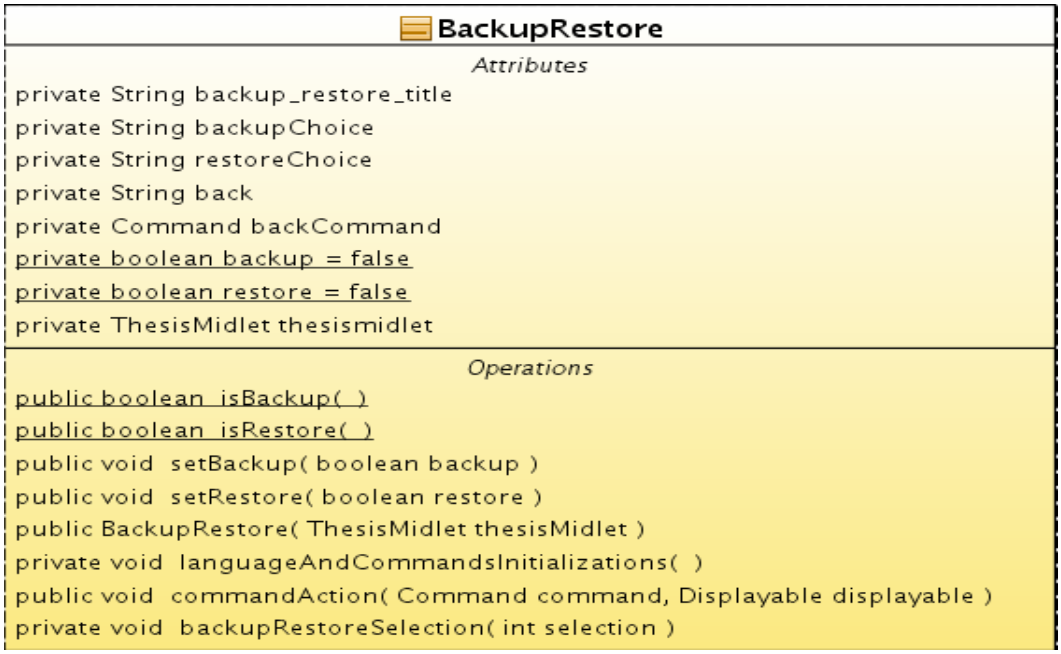
Οι επιλογές τις οποίες μας δίνει αυτή η class, είναι:

- Να κάνουμε backup ή restore τα Contacts
- Να κάνουμε backup ή restore τα Events
- Να κάνουμε backup ή restore τα ToDos

Όποιο από τα συστατικά μέρη του PIM κι αν διαλέξουμε, θα φορτωθεί η class FileSelector με την αντίστοιχη επιλογή, ενώ σε περίπτωση που ο χρήστης επιλέξει να πάει πίσω, τότε θα φορτωθεί η class ActionSelection

Ο έλεγχος όλων των παραπάνω επιλογών γίνεται με την χρήση της μεθόδου *private void pimSelection (int selection)* η οποία χρησιμοποιώντας την δομή επιλογής switch, μας δίνει το αποτέλεσματα μία ενέργειας, για την κάθε επιλογή.

Class BackupRestore



Εικόνα 9.6 - class BackupRestore

Η class BackupRestore είναι εκείνη μέσω της οποίας επιλέγουμε το τι θα κάνουμε μέσα στο πρόγραμμα. Αυτή η class κάνει extend την List και κάνει implement το CommandListener. Ανήκει στο package selections, καθώς ο χρήστης επιλέγει τα στοιχεία της, τα οποία εμφανίζονται στην οθόνη.

Ο constructor έχει σαν όρισμα το MIDlet, την ThesisMidlet, ενώ με την κλήση της μεθόδου *private void languageAndCommandsInitializations()* μπορεί και φορτώνει τις επιλογές για την γλώσσα που έχει επιλεχθεί.

Υπάρχουν οι μέθοδοι *public void setBackup(boolean flag)* και *public void setRestore(boolean restore)*

ώστε να κάνουμε set το αν έχει γίνει επιλογή για backup ή restore ενώ με τις μεθόδους *public static boolean isBackup()* και *public static boolean isRestore()* βλέπουμε την επιλογή του χρήστη.

Οι επιλογές τις οποίες μας δίνει αυτή η class, είναι:

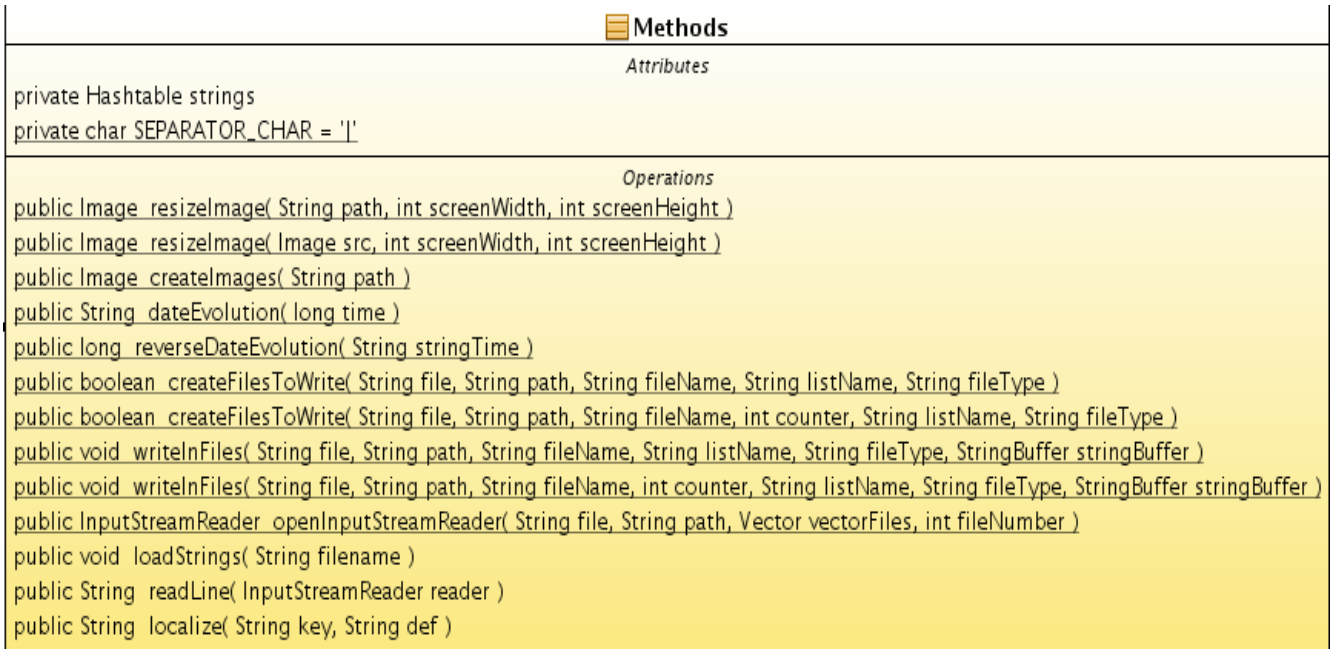
- Να κάνουμε backup των δεδομένων
- Να κάνουμε restore των δεδομένων
- Να πάμε πίσω την επιλογή και πάλι που μέρους του PIM θέλουμε

Σε περίπτωση που επιλέξουμε να κάνουμε backup, τότε θα φορτωθεί η class FileSelector, με την μέθοδο isBackup() να επιστρέφει true, ενώ στην επιλογή για restore θα φορτωθεί η class FileSelector με την μέθοδο isRestore να επιστρέφει true.

Ο έλεγχος όλων των παραπάνω επιλογών γίνεται με την χρήση της μεθόδου *private void backupRestoreSelection (int selection)* η οποία χρησιμοποιώντας την δομή επιλογής switch, μας δίνει το αποτέλεσματα μία ενέργειας, για την κάθε επιλογή.

9.2.3 Package utils

Class Methods



Εικόνα 9.7 - class Methods

Η class Methods όπως δηλώνει και το όνομά της, περιέχει κάποιες μεθόδους, οι οποίες καλούνται από τις διάφορες classes του προγράμματος. Ανήκει στο package utils, καθώς περιέχει χρήσιμες μεθόδους

για τις υπόλοιπες classes της εφαρμογής.

Η μέθοδος *public static Image resizeImage(String path, int screenWidth, int screenHeight)* δέχεται σαν παράμετρο το path ενός Image, το πλάτος της οθόνης, το ύψος της οθόνης κι επιστρέφει ένα Image, του οποίου έχει αλλάξει η ανάλυση, προκειμένου να χωράει στην οθόνη κάθε κινητής συσκευής.

Η μέθοδος *public static Image resizeImage(Image src, int screenWidth, int screenHeight)* δέχεται σαν παράμετρο ένα Image, το πλάτος της οθόνης, το ύψος της οθόνης κι επιστρέφει ένα Image, του οποίου έχει αλλάξει η ανάλυση, προκειμένου να χωράει στην οθόνη κάθε κινητής συσκευής.

Η μέθοδος *public Image createImages(String path)* δέχεται σαν παράμετρο μία το path ενός Image, το δημιουργεί κι επιστρέφει ένα Image.

Η μέθοδος *public static String dateEvolution(long time)* δέχεται σαν παράμετρο την long τιμή μίας ημερομηνίας κι επιστρέφει ένα String format της (το οποίο αναγνωρίζεται από αρχεία τύπου vCard και vCalendar).

Η μέθοδος *public static long reverseDateEvolution(String stringTime)* δέχεται σαν παράμετρο ένα String format ημερομηνίας (το οποίο αναγνωρίζεται από αρχεία τύπου vCard και vCalendar) κι επιστρέφει την long τιμή της.

Η μέθοδος *public static boolean createFilesToWrite(String file, String path, String fileName, String listName, String fileType)* χρησιμοποιείται όταν θέλουμε να δημιουργήσουμε αρχεία για να αποθηκεύσουμε δεδομένα που αφορούν contacts. Επιστρέφει μία boolean για το αν έχουν δημιουργηθεία καινούρια αρχεία ή όχι.

Η μέθοδος *public static boolean createFilesToWrite(String file, String path, String fileName, int counter, String listName, String fileType)* χρησιμοποιείται όταν θέλουμε να δημιουργήσουμε αρχεία για να αποθηκεύσουμε τα events και τα todos. Επιστρέφει μία boolean για το αν έχουν δημιουργηθεία καινούρια αρχεία ή όχι.

Η μέθοδος *public static void writeInFiles(String file, String path, String fileName, String listName, String fileType, StringBuffer stringBuffer)* χρησιμοποιείται όταν θέλουμε να αποθηκεύσουμε δεδομένα στα αρχεία τα οποία αφορούν τα contacts.

Η μέθοδος *public static void writeInFiles(String file, String path, String fileName, int counter, String listName, String fileType, StringBuffer stringBuffer*) χρησιμοποιείται όταν θέλουμε να αποθηκεύσουμε δεδομένα στα αρχεία τα οποία αφορούν τα events και τα todos.

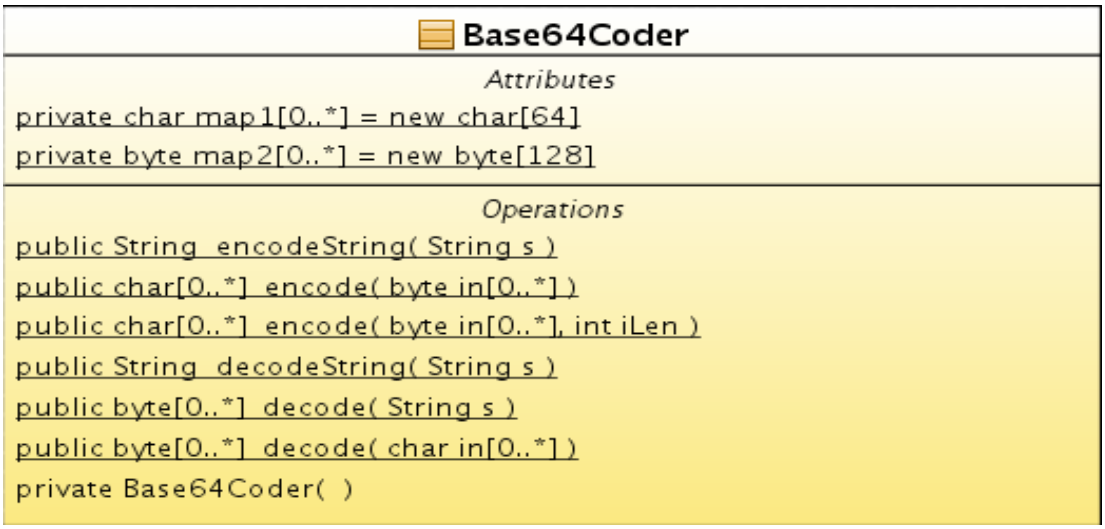
Η μέθοδος *public static InputStreamReader openInputStreamReader(String file, String path, Vector vectorFiles, int fileNumber)* χρησιμοποιείται όταν θέλουμε να διαβάσουμε δεδομένα από αρχεία τα οποία contacts, events ή todos. Επιστρέφει ένα InputStreamReader το οποίο διαβάζει μέσα στα αρχεία.

Η μέθοδος *public void loadStrings(String filename)* φορτώνει τα Strings τα οποία αφορούν τα αρχεία με τις διάφορες γλώσσες.

Η μέθοδος *public String readLine(InputStreamReader reader)* διαβάζει τα αρχεία με τα οποία αφορούν τις διάφορες γλώσσες κι επιστρέφει ένα String με το περιεχόμενο τους.

Η μέθοδος *public String localize(String key, String def)* παίρνει την φράση προς εμφάνιση κι επιστρέφει την μετάφρασή της.

Class Base64Coder



Εικόνα 9.8 - class Base64Coder

Η class Base64Coder κωδικοποιεί κι αποκωδικοποιεί τα binary και String δεδομένα μίας εικόνας επαφής. Ανήκει στο package utils, καθώς περιέχει χρήσιμες μεθόδους για τις υπόλοιπες classes της εφαρμογής.

Η μέθοδος *public static String encodeString (String s)* δέχεται σαν παράμετρο ένα String για να κωδικοποιηθεί σε Base64 format. Επιστρέφει ένα String.

Η μέθοδος *public static char[] encode (byte[] in)* δέχεται σαν παράμετρο ένα πίνακα τύπου byte για να κωδικοποιηθεί σε Base64 format. Επιστρέφει έναν πίνακα τύπου char.

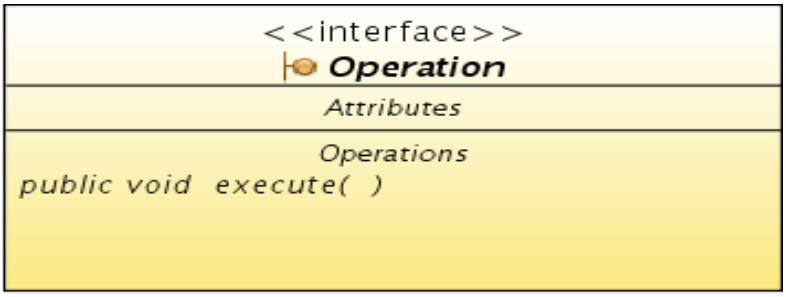
Η μέθοδος *public static char[] encode (byte[] in, int iLen)* δέχεται σαν παράμετρο ένα πίνακα τύπου byte, θέτοντας όμως το μέγεθος από τον πίνακα το οποίο θέλουμε να κωδικοποιηθεί σε Base64 format. Επιστρέφει έναν πίνακα τύπου char.

Η μέθοδος public static String decodeString (String s) δέχεται σαν παράμετρο ένα String για να αποκωδικοποιηθεί από Base64 format. Επιστρέφει ένα String.

Η μέθοδος *public static byte[] decode (String s)* δέχεται σαν παράμετρο ένα String για να αποκωδικοποιηθεί από Base64 format. Επιστρέφει έναν πίνακα τύπου byte.

Η μέθοδος *public static byte[] decode (char[] in)* δέχεται σαν παράμετρο ένα πίνακα τύπου char, τον πίνακα το οποίο θέλουμε να αποκωδικοποιηθεί από Base64 format. Επιστρέφει έναν πίνακα τύπου byte.

Interface Operation

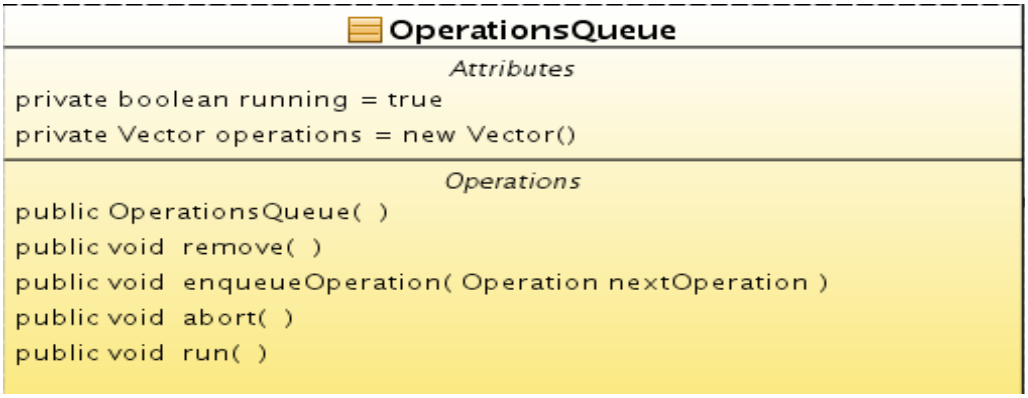


Εικόνα 9.9 - interface Operation

Το interface Operation είναι το interface μέσω του οποίου θα δημιουργηθεί η δομή στην οποία θα εισάγουμε τις επιλογές του χρήστη για την αλληλεπίδρασή της εφαρμογής με τα δεδομένα της κινητής συσκευής. Ανήκει στο package utils, είναι χρήσιμο για την σωστή λειτουργία του υπόλοιπου προγράμματος.

Έχει μία μόνο μέθοδο την *void execute()*

Class OperationsQueue



Εικόνα 9.10 - class ThesisMidlet

Η class OperationsQueue κάνει implement το interface Runnable κι έχει ως στόχο την ομαλή διαχείριση των λειτουργιών οι οποίες επιτελούνται από το πρόγραμμα. Στην ουσία είναι ένα thread για την παράλληλη εκτέλεση τους. Ανήκει στο package utils, καθώς περιέχει χρήσιμες μεθόδους για τις υπόλοιπες classes της εφαρμογής.

Η μέθοδος *void enqueueOperation(Operation nextOperation)* εισάγει σε ένα Vector τις λειτουργίες οι οποίες πρέπει να γίνουν για την εκτέλεση του πρόγραμματος.

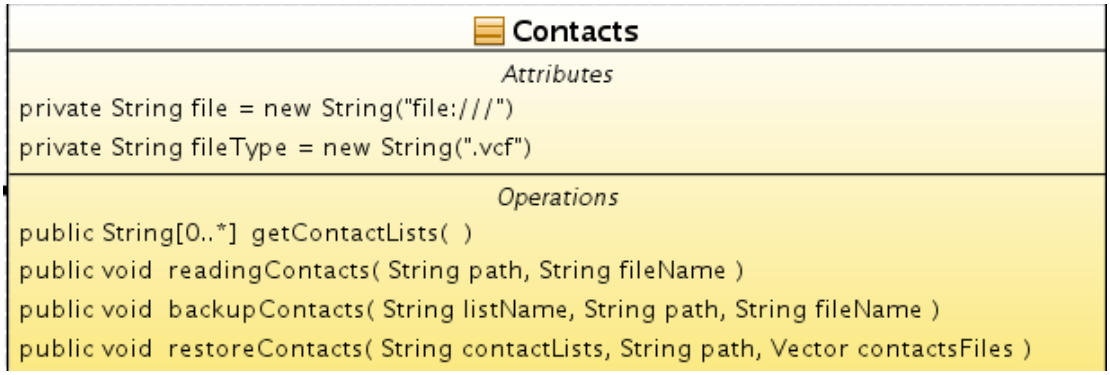
Η μέθοδος *void remove()* αφαιρεί από το Vector τις λειτουργίες του thread.

Η μέθοδος *void abort()* σταματάει την εκτέλεση του thread.

Η μέθοδος *run()* είναι η μέθοδος εκτέλεσης του thread.

9.2.4 Package pimComponents

Class Contacts



Εικόνα 9.11 - class Contacts

Η class Contacts περιέχει τις μεθόδους οι οποίες αφορούν τα contacts. Ανήκει στο package pimComponents, καθώς αποτελεί συστατικό μέρος (component) του PIM.

Η μέθοδος *public String [] getContactLists()* επιστρέφει μία λίστα από String η οποία περιέχει τις λίστες των contacts οι οποίες υπάρχουν στην κινητή συσκευή.

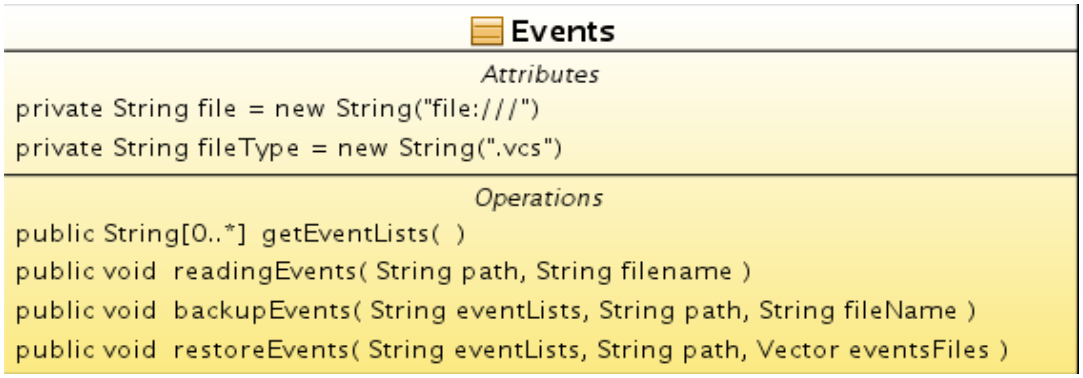
Η μέθοδος *public void readingContacts(String path, String fileName)* παίρνει ως πρώτη παράμετρο το σημείο στο οποίο θα αποθηκευεί το αρχείο με τα contacts, ενώ με την δεύτερη, το όνομα του αρχείου στο οποίο θα αποθηκεύσουμε τα contacts. Στο σώμα της μεθόδου αυτή, καλείται η public void

backupContacts(String listName, String path, String fileName) για κάθε διαφορετικό είδος contact list το οποίο υπάρχει στην κινητή συσκευή.

Η μέθοδος *public void backupContacts(String listName, String path, String fileName)* δημιουργεί το κάθε αρχείο για το backup των contacts. Η πρώτη παράμετρος είναι το όνομα του contact list στο οποίο γίνεται η προσπέλαση εκείνη την στιγμή, ενώ οι δύο επόμενες, είναι οι ίδιες με εκείνες οι οποίες αναφέρονται παραπάνω.

Η μέθοδος *public void restoreContacts(String contactLists, String path, Vector contactsFiles)* κάνει επαναφορά των contacts, ανοίγοντας κάθε αρχείο για κάθε contact list.

Class Events



Εικόνα 9.12 - class Events

Η class Events περιέχει τις μεθόδους οι οποίες αφορούν τα events. Ανήκει στο package pimComponents, καθώς αποτελεί συστατικό μέρος (component) του PIM.

Η μέθοδος *public String [] getEventLists()* επιστρέφει μία λίστα από String η οποία περιέχει τις λίστες των events οι οποίες υπάρχουν στην κινητή συσκευή.

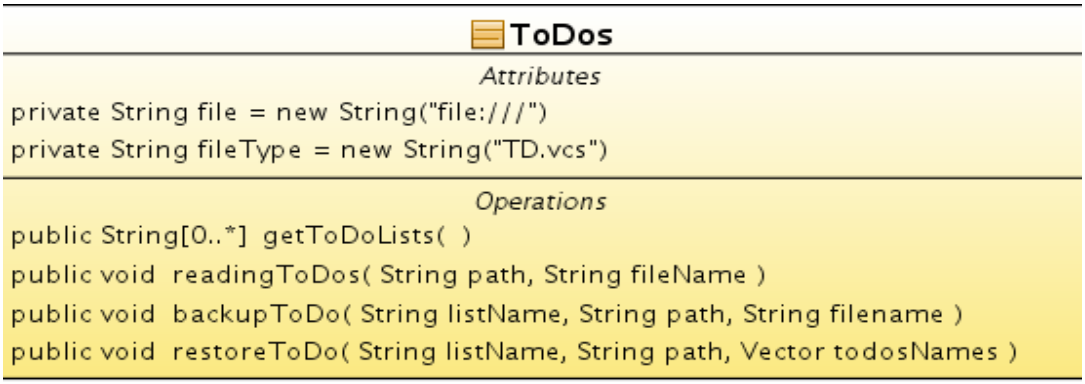
Η μέθοδος *public void readingEvents(String path, String fileName)* παίρνει ως πρώτη παράμετρο το σημείο στο οποίο θα αποθηκευεί το αρχείο με τα events, ενώ με την δεύτερη, το όνομα του αρχείου στο οποίο θα αποθηκεύσουμε τα events. Στο σώμα της μεθόδου αυτή, καλείται η public void backupEvents(String listName, String path, String fileName) για κάθε διαφορετικό είδος event list το οποίο υπάρχει στην κινητή συσκευή.

Η μέθοδος *public void backupEvents(String listName, String path, String fileName)* δημιουργεί το κάθε αρχείο για το backup των events. Η πρώτη παράμετρος είναι το όνομα του event list στο οποίο γίνεται η προσπέλαση εκείνη την στιγμή, ενώ οι δύο επόμενες, είναι οι ίδιες με εκείνες οι οποίες αναφέρονται παραπάνω.

Η μέθοδος *public void restoreEvents(String contactLists, String path, Vector contactsFiles)* κάνει

επαναφορά των events, ανοίγοντας κάθε αρχείο για κάθε event list.

Class ToDos



Εικόνα 9.13 - class ToDos

Η class ToDos περιέχει τις μεθόδους οι οποίες αφορούν τα events. Ανήκει στο package pimComponents, καθώς αποτελεί συστατικό μέρος (component) του PIM.

Η μέθοδος *public String [] getToDosLists()* επιστρέφει μία λίστα από String η οποία περιέχει τις λίστες των todos οι οποίες υπάρχουν στην κινητή συσκευή.

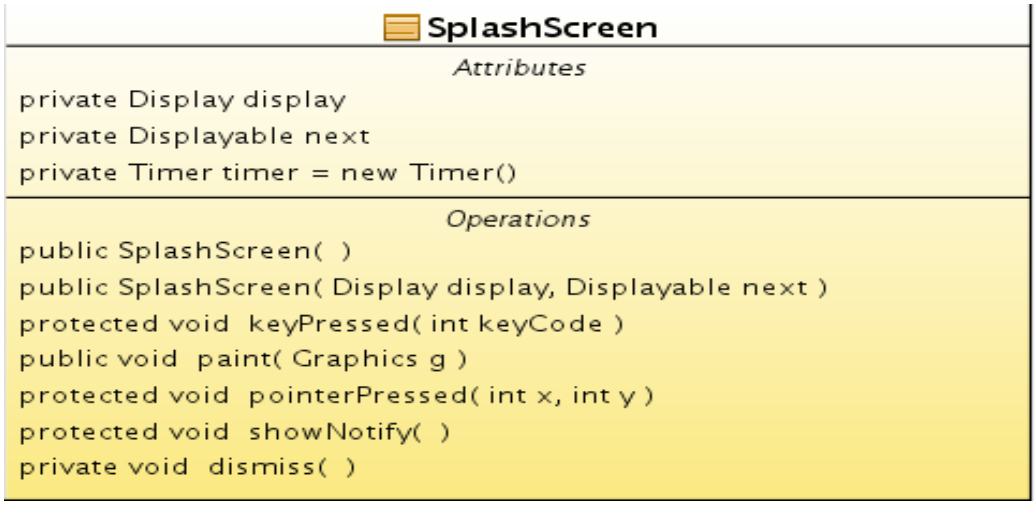
Η μέθοδος *public void readingToDos(String path, String fileName)* παίρνει ως πρώτη παράμετρο το σημείο στο οποίο θα αποθηκευεί το αρχείο με τα todos, ενώ με την δεύτερη, το όνομα του αρχείου στο οποίο θα αποθηκεύσουμε τα todos. Στο σώμα της μεθόδου αυτή, καλείται η public void backupEvents(String listName, String path, String fileName) για κάθε διαφορετικό είδος todo list το οποίο υπάρχει στην κινητή συσκευή.

Η μέθοδος *public void backupToDo(String listName, String path, String fileName)* δημιουργεί το κάθε αρχείο για το backup των todos. Η πρώτη παράμετρος είναι το όνομα του todo list στο οποίο γίνεται η προσπέλαση εκείνη την στιγμή, ενώ οι δύο επόμενες, είναι οι ίδιες με εκείνες οι οποίες αναφέρονται παραπάνω.

Η μέθοδος *public void restoreToDo(String contactLists, String path, Vector contactsFiles)* κάνει επαναφορά των events, ανοίγοντας κάθε αρχείο για κάθε todo list.

9.2.5 Package visual

Class SplashScreen



Εικόνα 9.14 - class SplashScreen

Η class SplashScreen κάνει extend την Canvas. Στην ουσία θα ζωγραφίσει μία εικόνα η οποία εμφανίζεται κατά την έναρξη του προγράμματος. Συνήθως τέτοιες εικόνες αναφέρουν κάποιες πληροφορίες για το πρόγραμμα ή κάποιες εικόνες οι οποίες θέλουν να εντυπωσιάσουν τον χρήστη και να τον προϊδεάσουν για το τι ακολουθεί. Ανήκει στο package visual, γιατί έχει μία οπτική (visual) σχέση με το υπόλοιπο πρόγραμμαα.

Ο constructor έχει σαν παράμετρο το Display το οποίο υπάρχει εκείνη την στιγμή στην οθόνη, δηλαδή το splashscreen και το Displayable, αυτό το οποίο θα ακολουθήσει μετά το τέλος του.

Η μέθοδος *private void dismiss()* σταματάει την απεικόνιση της εικόνας και προχωράει στο κυρίως πρόγραμμα. Αυτό γίνεται είτε με την παρέλευση κάποιου χρονικού διαστήματος, το οποίο έχει οριστεί hard coded είτε με την επιλογή του χρήστη να πατήσει κάποιο πλήκτρο.

Η μέθοδος *protected void pointerPressed(int x, int y)* αντιλαμβάνεται το πάτημα του joystic.

Η μέθοδος *protected void keyPressed(int keyCode)* αντιλαμβάνεται το πάτημα κάποιου πλήκτρου.

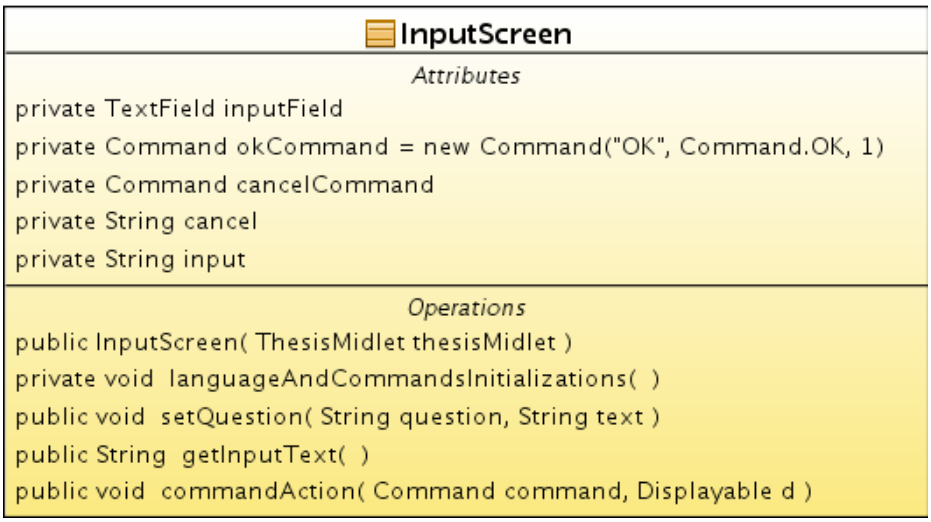
Η μέθοδος *protected void showNotify()* θέτει το χρονόμετρο το οποίο θα μετρήσει τον χρόνο κατά τον οποίο θα διαρκέσει η απεικόνιση στην οθόνη του SplashScreen.

Η μέθοδος *public void paint(Graphics g)* ζωγραφίζει στην ουσία την εικόνα η οποία θα είναι το SplashScreen.

Η class *private class CountDown* είναι μία private class η οποία κάνει extend την TimerTask. Στην ουσία είναι η class η οποία θα κρατήσει τον χρόνο για την διάρκεια απεικόνισης του SplashScreen.

Η μέθοδος *run()* είναι η μέθοδος η οποία με την παρέλευση του χρόνο απεικόνισης του SplashScreen, θα το σταματήσει για να συνεχιστεί το πρόγραμμα.

Class InputScreen



Εικόνα 9.15 - class InputScreen

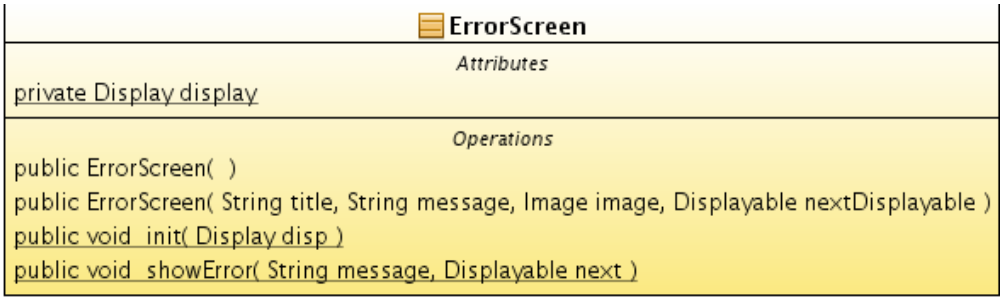
Η class InputScreen εμφανίζει στην οθόνη ένα textfield στο οποίο δίνουμε το όνομα ενός αρχείου ή ενός φακέλου. Η class αυτή κάνει extend την Form και implement το CommandListener. Ανήκει στο package visual, γιατί έχει μία οπτική (visual) σχέση με το υπόλοιπο πρόγραμμαα.

Ο constructor έχει σαν όρισμα το MIDlet, την ThesisMidlet, ενώ με την κλήση της μεθόδου *private void languageAndCommandsInitializations()* μπορεί και φορτώνει τις επιλογές για την γλώσσα που έχει επιλεχθεί.

Η μέθοδος *public void setQuestion(String question, String text)* στην ουσία καθοδηγεί τον χρήστη για το αν θέλει να εισάγει όνομα αρχείου ή φακέλου.

Η μέθοδος *public String getInputText()* επιστρέφει ένα String με το κείμενο το οποίο πληκτρολόγησε ο χρήστης σαν όνομα.

Class ErrorScreen



Εικόνα 9.16 - class ErrorScreen

Η class ErrorScreen θα εμφανίζει στην οθόνη διάφορα μηνύματα σχετικά με σφάλματα τα οποία μπορεί να προκύψουν κατά την διάρκεια του προγράμματος. Η class αυτή κάνει extend την Alert. Ανήκει στο package visual, γιατί έχει μία οπτική (visual) σχέση με το υπόλοιπο πρόγραμμαα.

Ο πρώτος constructor είναι default και υποδεικνείει μηνύματα λάθους.

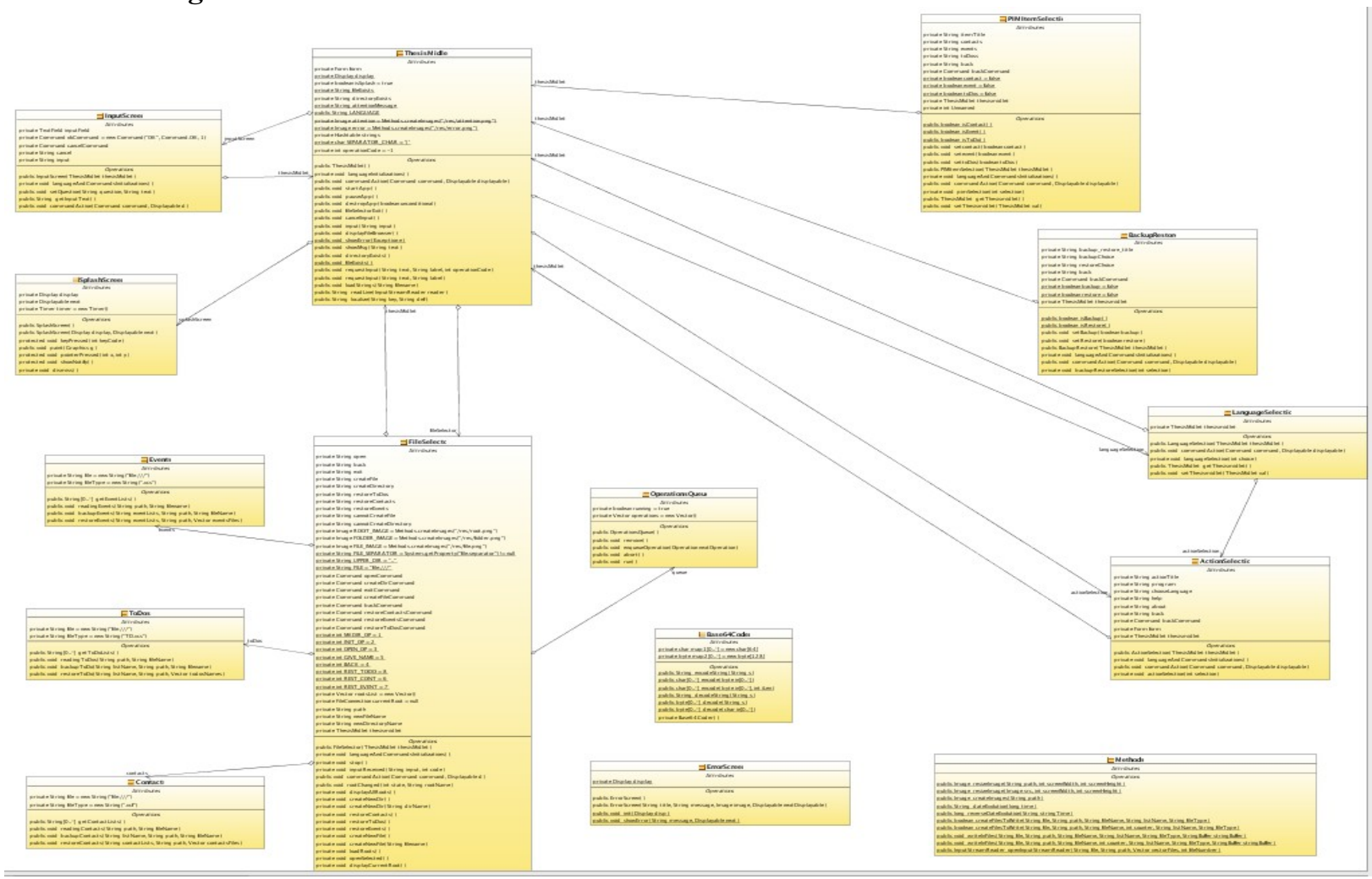
Ο δεύτερος constructor παίρνει παραμέτρους τον τίτλο της οθόνης την οποία θα εμφανίσει, το μήνυμα το οποίο θα διαβάσει ο χρήστης, την εικόνα η οποία θα τραβήξει την προσοχή, καθώς το επόμενο Displayable το οποίο θα φανεί στην οθόνη.

Η μέθοδος *public static void init(Display disp)* αρχειοκοποιεί το display της εφαρμογής.

Η μέθοδος *public static void showError(String message, Displayable next)* εμφανίζει σφάλματα τα οποία συμβαίνουν κατά την διάρκεια εκτέλεσης του προγράμματος.

9.3 Diagrams

9.3.1 Class Diagram

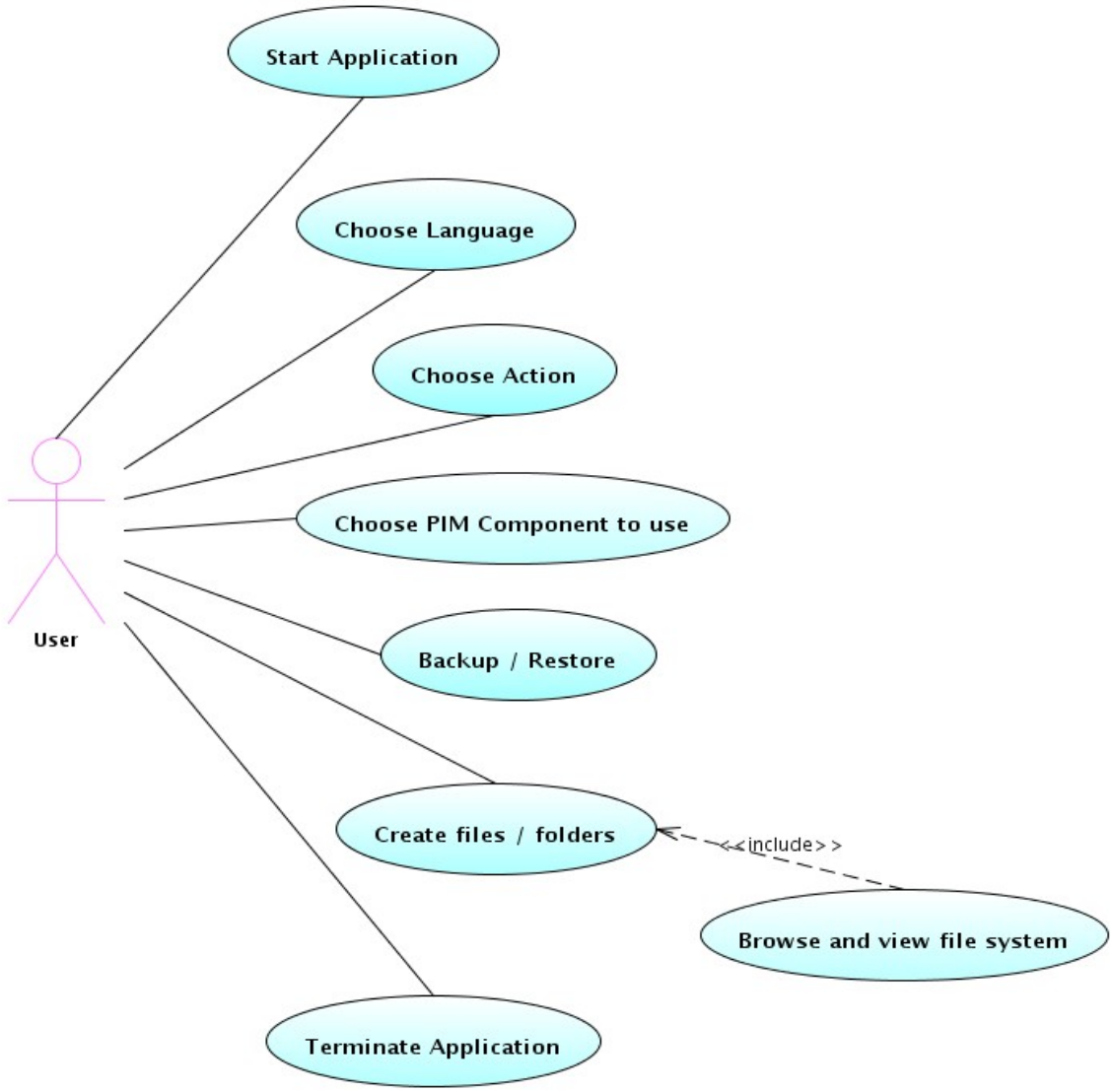


Εικόνα 9.17 - Class Diagram

Επειδή το class diagram ήταν πολύ μεγάλο, παραπάνω φαίνεται μία απεικόνισή του, για να καταλάβει ο χρήστης πως συνδέονται οι classes μεταξύ τους. Με τα διαγράμματα που ακολουθούν, οι όποιες απορίες θα ξεκαθαριστούν.

9.3.2 Use Case diagram

Με αυτό το Use Case διάγραμμα μπορούμε να δούμε τις ενέργειες του χρήστη προς το σύστημα. Ο χρήστης μπορεί να ξεκινήσει την εφαρμογή, να επιλέξει γλώσσα, να επιλέξει το τι θέλει να κάνει στο πρόγραμμα, να διαλέξει με ποιο από τα συστατικά μέρη του PIM θέλει να ασχοληθεί, αν θέλει να δημιουργήσει αντίγραφα ασφαλείας, ή να επαναφέρει ήδη υπάρχοντα αρχεία με δεδομένα, ενώ εμμέσως βλέπει το file system της κινητής συσκευής, ενώ τέλος, μπορεί να τερματίσει την εφαρμογή.

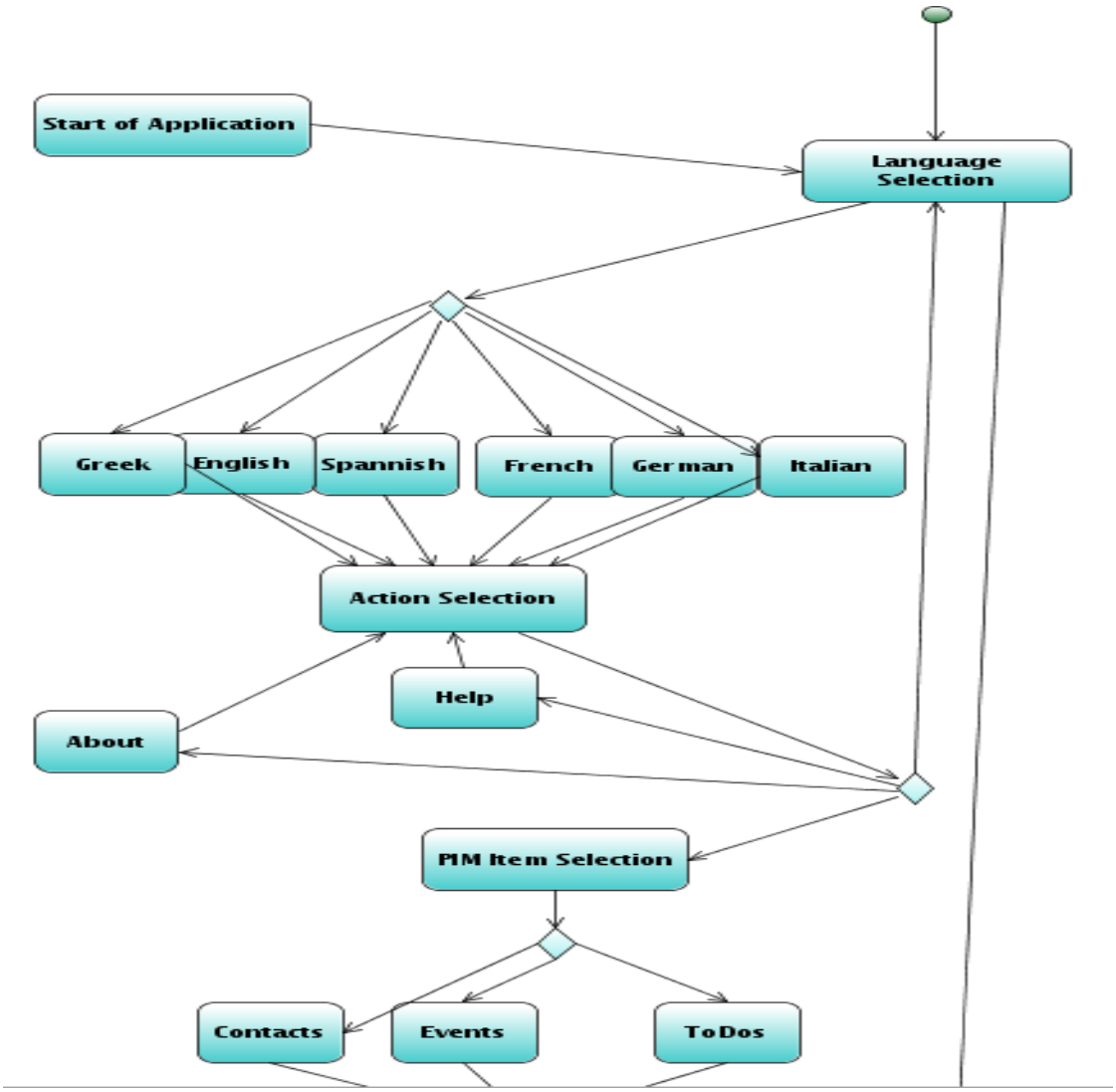


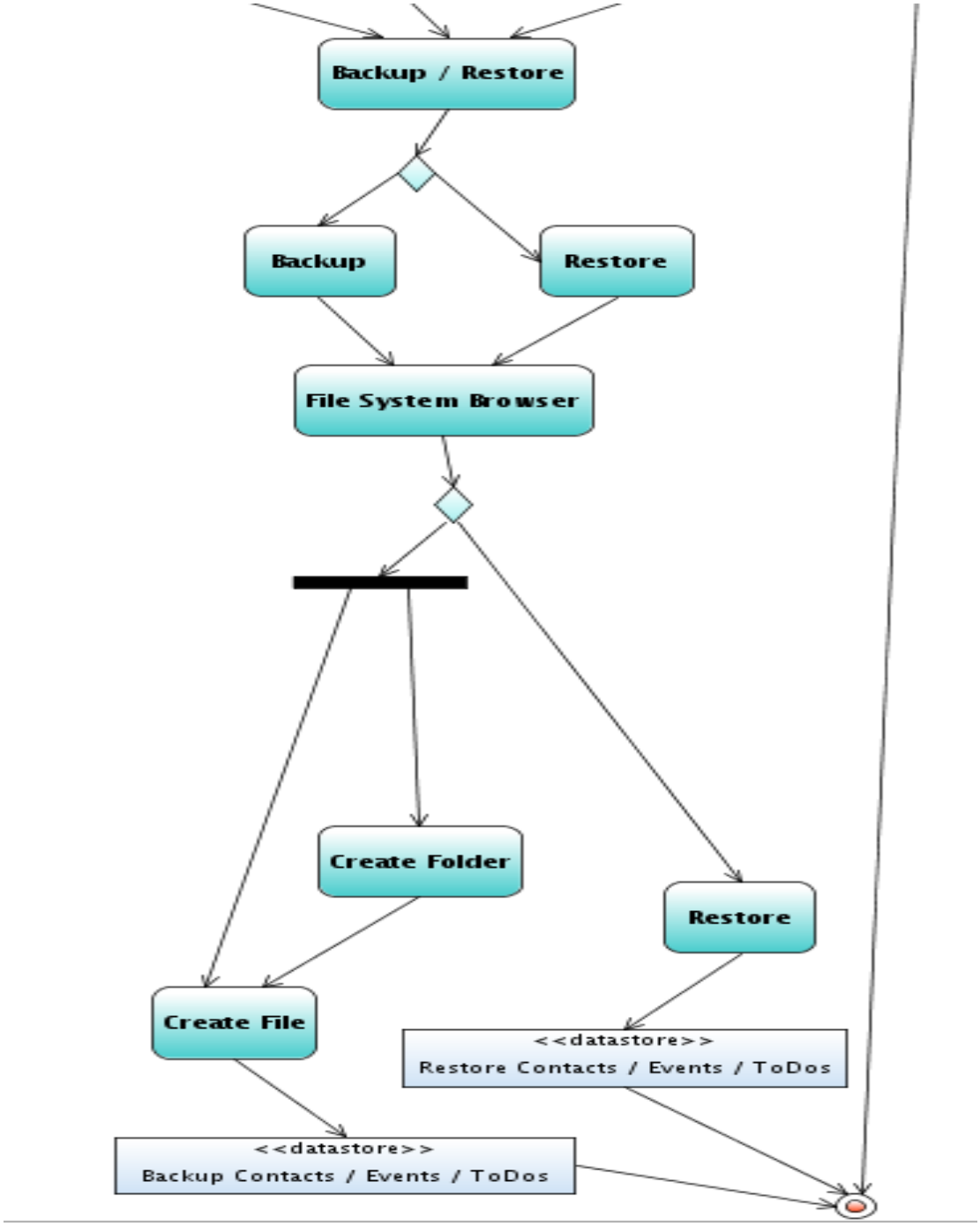
Εικόνα 9.18 - Use Case Diagram

9.3.3 Activity Diagram

Με το διάγραμμα αυτό μπορεί να δει ο αναγνώστης την ροή του προγράμματος από την έναρξή του μέχρι το τέλος του, μέσα από τις διάφορες καταστάσεις από τις οποίες περνάει η εκτέλεση του.

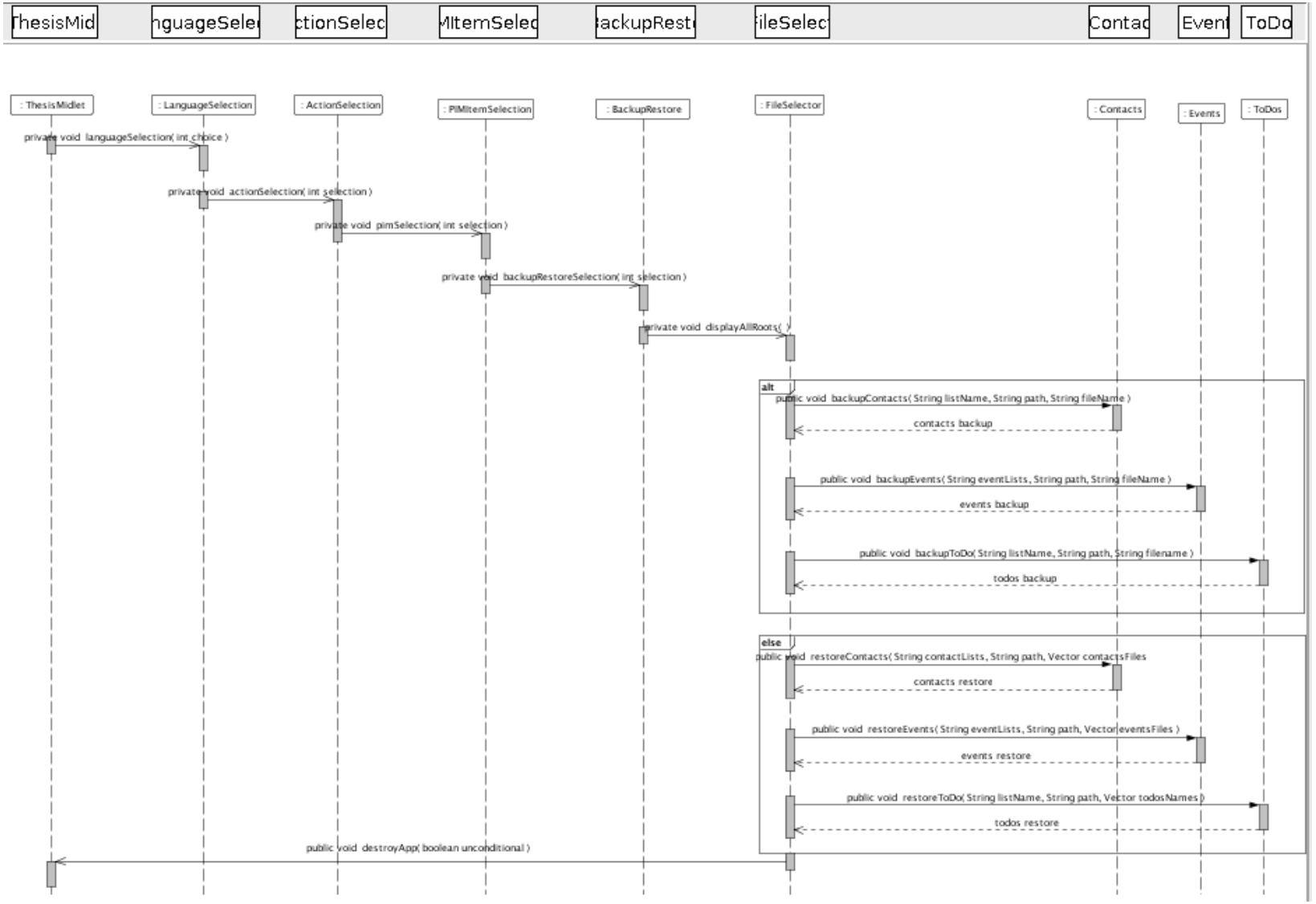
Με την έναρξη του προγράμματος, ο χρήστης θα πρέπει να επιλέξει την γλώσσα της αρεσκείας του, η οποία μπορεί να είναι Ελληνικά, Αγγλικά, Ισπανικά, Γαλλικά, Γερμαντικά, Ιταλικά. Έπειτα θα πρέπει να επιλέξει το τι θέλει να κάνει στο πρόγραμμα. Μπορεί αν θέλει να ξαναδιαλέξει γλώσσα, να δει την βοήθεια ως προς την χρήση του προγράμματος ή να δει πληροφορίες για αυτό. Στην συνέχεια διαλέγει με πιο από τα 3 PIM συστατικά μέρη θέλει να ασχοληθεί ενώ αμέσως μετά επιλέγει αν θέλει να κρατήσει αντίγραφα ασφαλείας ή να κάνει επαναφορά. Αν επιλέξει την δημιουργία αντιγράφων ασφαλείας, τότε μπορεί να δημιουργήσει ένα φάκελο και μέσα τα αντίγραφα, είτε μπορεί να δημιουργήσει κατευθείαν το αρχείο με τα δεδομένα και ύστερα μπορεί να επιλέξει τερματιστμό της εφαρμογής ή την συνέχιση της με κάποια άλλη ενέργεια. Αν επιλέξει την επαναφορά των δεδομένων, ύστερα μπορεί να τερματίσει την εφαρμοή, ή να συνεχίσει με κάποια άλλη ενέργεια.





Εικόνα 9.19 - Activity Diagram

9.3.4 Sequence Diagram



Εικόνα 9.20 - Sequence Diagram

Στο παραπάων διάγραμμα ακολουθίας ο αναγνώστης μπορεί να δει την ακολουθία την οποία ακολουθεί η εφαρμογή. Μετά την επιλογή της γλώσσας, ακολουθεί έπειτα η επιλογή του τι θέλει να κάνει ο χρήστης στο πρόγραμμα, μετά η επιλογή ποιου συστατικού μέρους του PIM θέλει να επεξεργαστεί, ύστερα η επιλογή του αν θα δημιουργήσει αντίγραφα ασφαλείας ή επαναφορά και τέλος μετά την πλοήγηση στο file system, η διεκπαιρέωση της επιθυμητής λειτουργίας και ο τερματισμός του προγράμματος.

9.4 Ανακεφαλαίωση

Σε αυτό το κεφάλαιο ο αναγνώστης είδε την UML πλευρά του προγράμματος με την χρήση διαφόρων διαγραμμάτων και εικόνων. Στο επόμενο κεφάλαιο παρουσίαζεται το NetBeans IDE με το οποίο έγινε η ανάπτυξη του κώδικα.

Κεφάλαιο 10ο

10.1 Εισαγωγή

Σε αυτό το κεφάλαιο γίνεται μία μικρή παρουσίαση των χαρακτηριστικών του NetBeans IDE με την χρήση του οποίου έγινε η ανάπτυξη του κώδικα, αλλά των UML διαγραμμάτων.

Εικόνα 10.1 - Netbeans logo

10.2 Εισαγωγή στο NetBeans

Το NetBeans αναφέρεται ως η πλατφόρμα για την ανάπτυξη επιτραπέζιων εφαρμογών Java αλλά και ως ένα ολοκληρωμένο περιβάλλον ανάπτυξης (IDE) το οποίο αναπτύχθηκε χρημιμοποιώντας την πλατφόρμα NetBeans.

Η πλατφόρμα NetBeans επιτρέπει την ανάπτυξη εφαρμογών οι οποίες αναπτύσσονται από ένα πρότυπο συναρτησιακών στοιχείων συνιστόμενων μερών λογισμικού, τα οποία ονομάζονται πρότυπα συναρτησιακά στοιχεία (modules). Ένα module είναι ένα Jar αρχείο το οποίο περιέχει τις Java κλάσσεις οι οποίες έχουν γραφτεί για να αλληλεπιδρούν με τα Open APIs του NetBeans και ένα αρχείο manifest το οποίο το ταυτοποιεί σαν ένα module. Εφαρμογές δημιουργημένες σε modules μπορούν να επεκταθούν προσθέτοντας καινούρια modules. Αφού τα modules μπορούν να αναπτυχθούν ανεξάρτητα, βασισμένα στην πλατφόρμα NetBeans μπορούν να επεκταθούν κι από τρίτους κατασκευαστές λογισμικού.

10.3 Ιστορική Αναδρομή

10.3.1 Τα πρώτα χρόνια

Το NetBeans ξεκίνησε το 1997 σαν Xelfi, ένα φοιτητικό project κάτω από την καθοδήγηση του Τμήματος Μαθηματικών και Φυσικής στο Charles University στην Πράγα. Ο Roman Stanek αργότερα δημιούργησε μία εταιρεία γύρω από αυτό το project και παρήγαγε εμπορικές εκδόσεις του NetBeans IDE μέχρι που αγοράστηκε από την Sun Microsystems το 1999. Η Sun άνοιξε τον κώδικα του NetBeans IDE το Ιούνιο του επόμενου χρόνου.

10.3.2 Τωρινές εκδόσεις

Το NetBeans IDE 6.1 επέκτεινε τα υπάρχοντα χαρακτηριστικά της Java Enterprise Edition (Java EE).

Επιπροσθέτως το NetBeans Enterprise Pack υποστηρίζει την ανάπτυξη εταιρικών εφαρμογών με την την Java EE 5, συμπεριλαμβανουμένου.

Το NetBeans 6.0 βασίστηκε στην προηγούμενη του έκδοση 5.5.1, η οποία εισήγαγε εκτενή υποστήριξη για την ανάπτυξη IDE modules και πολλές client εφαρμογές βασισμένες στην πλατφόρμα NetBeans, ένα καινούριο GUI περιβάλλον, νέα και επανασχεδιασμεη CVS υποστήριξη, Weblogic 9 και JBoss 4 υποστήριξη και πολλές βελτιώσεις στον editor. Το NetBeans 6.0 προσφέρεται σαν μέρος της διανομής Linux Ubuntu 8.04.

NetBeans IDE 6.1 extended the existing Java EE features (including Java Persistence support, EJB 3 and JAX-WS). Additionally, the NetBeans Enterprise Pack supports development of Java EE 5 enterprise applications, including SOA visual design tools, XML schema tools, web services orchestration (for BPEL), and UML modeling. The NetBeans C/C++ Pack supports C/C++ projects.

10.4 Η NetBeans Platform

Η πλατφόρμα NetBeans είναι ένα επαναχρησιμοποιούμενο framework για την απλοποίηση της ανάπτυξης άλλων επιτραπέζιων εφαρμογών. Όταν εκτελείται μία εφαρμογή η οποία βασίζεται στην πλατφόρμα NetBeans, η Κύρια κλάση της πλατφόρμας εκτελείται. Διαθέσιμα modules εντοπίζονται, τοποθετούνται σε μία registry μνήμης και οι διεργασίες των modules εκτελούνται. Γενικά, ο κώδικας ενός module φορτώνεται στην μνήμη μόνο όπως αυτό χρειάζεται.

Εφαρμογές μπορούν να εγκαταστήσουν modules δυναμικά. Κάθε εφαρμογή μπορεί να συμπεριλάβει το Update Center module ώστε να επιτρέπει στους χρήστες της εφαρμογής να κάνουν download σε ψηφιακά υπογεγραμμένες αναβαθμίσεις και καινούρια χαρακτηριστικά απευθείας στην εφαρμογή υπό εκτέλεση. Επαναεγκαθιστώντας μία αναβάθμιση ή μία νέα κυκλοφορία δεν αναγκάζει τους χρήστες να κατεβάσουν την όλη εφαρμογή ξανά.

Η πλατφόρμα προσφέρει υπηρεσίες κοινές με επιτραπέζιες εφαργογές, επιτρέποντας στους προγραμματιστές να επικεντρωθούν συγκεκριμένς στην λογική της δικής τους εφαρμογής. Ανάμεσα στα χαρακτηριστικά της πλατφόρμας είναι:

Διαχείριση διασύνδεσης χρήστη (πχ μενού και toolbars)

Διαχείρηση ρυθμίσεων χρήστη

Διαχείριση αποθήκευσης (αποθήκευση και φόρτωμα κάθε είδους δεδομένων)

Διαχείρηση παραθύρων

Wizard framework (υποστηρίζει βήμα προς βήμα μηνύματα)

10.5 Το NetBeans IDE

Το NetBeans IDE είναι ένα ανοιχτού κώδικα ολοκληρωμένο περιβάλλον ανάπτυξης γραμμένο εξ ολοκλήρου σε Java, χρησιμοποιώντας την πλατφόρμα NetBeans. Το NetBeans IDE υποστηρίζει ανάπτυξη όλων των τύπων εφαρμογές Java (J2SE, J2ME, web, EJB εφαρμογές). Ανάμεσα σε άλλα

χαρακτηριστικά, είναι και τα Ant-based project συστήματα, έλεγχοι έκδοσης και refactoring.

Modules: Όλες οι λειτουργείες του IDE προσφέρονται από τα modules. Κάθε module προσφέρει μία καλά καθορισμένη λειτουργία, όπως υποστήριξη για την γλώσσα Java, επεξεργασία, ή υποστήριξη για το Ταυτόχρονο Σύστημα εκδόσεων (CVS). Το NetBeans περιέχει όλα εκείνα τα modules για την ανάπτυξη Java με ένα μοναδικό download, επιτρέποντας στον χρήστη να ξεκινήσει αμέσως. Τα modules ακόμα, επιτρέπουν στο NetBeans να επεκταθεί. Νέα χαρακτηριστικά όπως η υποστήριξη για άλλες γλώσσες προγραμματισμού, μπορούν να προστεθούν, εγκαθιστώντας επιπρόσθετα modules.

Άδεια: Από τον Ιούλιο του 2006 μέχρι 2007, το NetBeans IDE είχε την άδεια Άδεια Κοινής Ανάπτυξης και Διανομής (CDDL) της Sun, μία άδεια η οποία είναι βασισμένη στην Γενική Άδεια Mozilla (MPL). Τον Οκτώβριο του 2007, η Sun ανακοίνωσε ότι το NetBeans εφεξής θα διατείθεται υπό την διπλή άδεια της CCDL και της GPL έκδοση 2, με την εξαίρεση από την GPL για το GNU classpath.

10.6 Ολοκληρωμένα modules για το NetBeans

10.6.1 Το NetBeans Profiler

Το NetBeans Profiler είναι ένα εργαλείο για την παρακολούθηση των εφαρμογών Java: Βοηθάει στο να βρεις κάποιος τα memory leaks και να βελτιστοποιήσει την ταχύτητα. Ενώ παλαιότερα ήταν ξεχωριστό κομμάτι, από την έκδοση 6.0. έχει ενσωματωθεί στο βασικό IDE.

Το Profiler βασίζεται σε ένα ερευνητικό πρόγραμμα των Sun Laboratories με το όνομα JFluid. Η έρευνα εκείνη αποκάλυψε συγκεκριμένες τεχνικές οι οποίες μπορούν να χρησιμοποιηθούν για τη μείωση του overhead σε μια εφαρμογή Java. Μία από αυτές τις τεχνικές είναι το δυναμικό bytecode instrumentation, το οποίο είναι ιδιαίτερα χρήσιμο για το profiling μεγάλων εφαρμογών Java. Χρησιμοποιώντας δυναμικό bytecode instrumentation και επιπλέον αλγορίθμους, το NetBeans Profiler είναι σε θέση να λάβει πληροφορίες σχετικά με το χρόνο εκτέλεσης των εφαρμογών οι οποίες είναι πολύ μεγάλες ή πολύπλοκες για άλλους profilers. Το NetBeans υποστηρίζει επίσης τα profiling points, τα οποία οποία επιτρέπουν το profiling σε ακριβή σημεία της εκτέλεσης και μέτρηση του χρόνου εκτέλεσης.

10.6.2 GUI σχεδιαστικό εργαλείο

Μέχρι πρότεινως γνωστό σαν project Matisse, το εργαλείο σχεδιάσης GUI επιτρέπει στους προγραμματιστές να πρωτοτυπούν και να σχεδιάζουν Swing GUI απλά τραβώντας και θέτωντας μέρη του GUI στην οθόνη του υπολογιστή τους. Ο δημιουργός GUI έχει υποστήριξη και για τα JSR 296 (Swing Application Framework) και JSR 295 (Beans Binding τεχνολογία).

10.6.3 NetBeans IDE για Mobile εφαρμογές

Το NetBeans IDE Bundle για κινητά είναι ένα εργαλείο για την ανάπτυξη εφαρμογών οι οποίες τρέχουν σε συσκευές κινητής. Γενικά κινητής τηλεφωνίας, αλλά αυτό περιλαμβάνει και PDA μεταξύ άλλων. Το Πακέτο Κινητής (Mobility Pack) μπορεί να χρησιμοποιηθεί για να γράψεις, να δοκιμάσεις και να αποσφαλματώσεις εφαργμογές για την πλατφόρμα Java Micro Edition (Java ME platform).

Ενσωματώνει υποστήριξη για το Mobile Information Device Profile (MIDP) 2.0, το Connected Limited Device Configuration (CLDC) 1.1 και το Connected Device Configuration (CDC). Κάποιος μπορεί εύκολα να ενσωματώσει προγράμματα κι από τρίτους για ένα στιβαρό περιβάλλον ελέγχου. Κατά την παρούσα περίοδο το πακέτο είναι διαθέσιμο σαν δύο ξεχωριστές κατηγορίες. Η μία καλύπτει CDC και η άλλη CLDC συσκευές.

[52]

10.7 Ανακεφαλαίωση

Σε αυτό το κεφάλαιο ο αναγνώστης είδε κάποια χαρκατηριστικά του NetBeans IDE. Στο επόμενο κεφάλαιο γίνεται αναφορά στα προβλήματα τα οποία αντιμετωπίστηκαν κατά την ανάπτυξη της εφαρμογής.

Κεφάλαιο 11ο

11.1 Εισαγωγή

Σε αυτό το κεφάλαιο γίνεται μία παρουσίαση των προβλημάτων τα οποία παρουσίαστηκαν κατά την ανάπτυξη της εφαρμογής. Κοινή πηγή του κακού; Ο κατακερματιστμός ο οποίος αναφέρθηκε και σε προηγούμενο κεφάλαιο.

11.2 Προβλήματα κατά την διάρκεια ανάπτυξης της πτυχιακής

Κατά την διάρκεια ανάπτυξης αυτής της πτυχιακής υπήρξαν αρκετά προβλήματα όσον αφορά στην δημιουργία του προγράμματος.

Το πιο σημαντικό από όλα, ήταν ο κατακερματισμός εξαιτίας του οποίο το πρόγραμμα λειτουργεί θεωρητικά σε όλες τις κινητές συσκευές, αλλά επιβεβαιωμένα σε κάποια μοντέλα Nokia μόνο. Έγιναν προσπάθειες εκτέλεσης της και σε συσκευές, Sony Ericsson, χωρίς όμως πλήρη επιτυχία. Κύριο σημείο αναφοράς είναι η διαφορά του file system που χρησιμοποιεί η κάθε μία εταιρεία για τις συσκευές της.

Ένα άλλο πρόβλημα, ήταν η μετατροπή της εφαρμογής σε πολύγλωσση. Μπορεί η ίδια η J2ME να έχει το JSR 238 το οποίο και να λύνει τα θέματα του localization, όμως παρ'όλα αυτά, υπάρχουν αρκετά προβλήματα ακόμα. Το JSR 238 από μόνο του, δίνει τις απαραίτητες βιβλιοθήκες, όμως η μετατροπή όλων των αρχείων με τις διάφορες γλώσσες σε binary αρχεία, απαιτεί την χρήση ειδικών εργαλείων, τα οποία βέβαια προσφέρει η Nokia [57, 58], καθώς και το API για την χρήση του. Όμως η ίδια πάλι η εταιρεία, δεν έχει συσκευές οι οποίες να υποστηρίζουν το εν λόγω JSR. Οπότε στην ουσία, από το 2005 προσφέρει ένα εργαλείο, το οποίο μέχρι και το 2009 είναι λίγο πολύ άχρηστο. Αξίζει να σημειωθεί ακόμα, πως η χρήση του εργαλείου αυτού, απαιτεί και την γνώση ant script, xml και την δημιουργία ενός ολόκληρου project για την καλύτερη χρήση του.

Ένα άλλο εμπόδιο είναι η περιορισμένη βιβλιογραφία. Η πιο πρόσφατες εκόδσεις είναι τουλάχιστον 4 ετών, πράγμα εξωφρενικό για μία γλώσσα η οποία εξελίσσεται συνεχώς, ενώ στην Ελλάδα δεν υπάρχουν βιβλία, και το Internet έχει πολύ περιορισμένες πληροφορίες και αρκετές φορές λανθασμένες για το αντικείμενο. Τα προγράμματα παραδείγματα από τις διάφορες ιστοσελίδες είναι προβληματικά και δεν τρέχουν χωρίς την παρέμβαση στον κώδικα. Ακόμη πολλές φορές πληροφορίες οι οποίες υπάρχουν σε διάφορα σημεία, έχουν δυστυχώς διαδωθεί εσφαλμένες, δυσκολεύοντας ακόμα περισσότερο την αναζήτηση του ενδιαφερόμενου.

11.3 Ανακεφαλαίωση

Σε αυτό το κεφάλαιο έγινε ανάλυση πάνω στα προβλήματα τα οποία εμφανίστηκαν κατά την δημιουργία αυτή της εργασίας. Στο επόμενο κεφάλαιο, θα δούμε τις καινούριες τεχνολογίες οι οποίες αναμένονται με αγωνία το 2009 και 2010.

Κεφάλαιο 12

12.1 Εισαγωγή

Σε αυτό το κεφάλαιο, το οποίο είναι και το τελευταίο, γίνεται μία μικρή παρουσίαση των καινούριων τεχνολογιών, οι οποίες θα κάνουν μεγάλη αίσθηση τις χρονιές 2009 και 2010 και αναμένονται από όλους με μεγάλη ανυπομονησία. Περισσότερες εφαρμογές, καλύτεροι και γρηγορότεροι τρόποι επικοινωνίας, είναι μερικές από τις τεχνολογίες οι οποίες θα αναφερθούν παρακάτων.

12.2 Νέες τεχνολογίες που αναμένονται το 2009 - 2010

Bluetooth 3.0

Η προδιαγραφή του Bluetooth 3.0 θα κυκλοφορήσει το 2009 με τις συσκευές οι οποίες θα το χρησιμοποιούν, να αρχίζουν να φθάνουν περίπου το 2010. Το Bluetooth 3.0 πιθανόν θα περιλαμβάνει χαρακτηριστικά όπως εξαιρετικά χαμηλή κατανάλωση, κάτι το θα επιτρέψει νέες συσκευές, όπως τα περιφερειακά και αισθητήρες, καθώς και νέες εφαρμογές, όπως η παρακολούθηση της υγείας. Το Bluetooth ξεκίνησε ως ένα σύνολο πρωτοκόλλων που λειτουργούν πάνω από μία μόνο ασύρματη τεχνολογία στον κομιστή. Το Bluetooth 3.0 προορίζεται να στηρίξει τρεις φορείς: "κλασικό" Bluetooth, Wi-Fi και ultrawideband (UWB). Είναι πιθανό ότι περισσότεροι φορείς θα υποστηριχθούν στο μέλλον.

Mobile User Interfaces (UIs)

Το user interface έχει σημαντική επίδραση στην ευχρηστία της συσκευής και στην υποστήριξή της. Επίσης, θα είναι μια περιοχή του έντονου ανταγωνισμού για το 2009 και το 2010, με τους κατασκευαστές να χρησιμοποιούν user interfaces για να διαφοροποιήσουν τις συσκευές και τις πλατφόρμες τους. Νέα και πιο διαφοροποιημένα user intrefaces θα περιπλέξουν την ανάπτυξη και την υποστήριξη των επιχειρήσεων προς τους εργαζομένους (B2E) και των επιχειρήσεων προς τους καταναλωτές (B2C) εφαρμογές. Οι οργανισμοί θα πρέπει να περιμένουν περισσότερες απαιτήσεις από τους χρήστες για την υποστήριξη των συγκεκριμένων μοντέλων καθοδηγούμενες από τις προτιμήσεις του interface.

Location Sensing

Το location awareness κάνει τις εφαρμογές κινητών συσκευών πιο ισχυρές και χρήσιμες. Στο μέλλον, η θέση θα αποτελέσει βασικό συστατικό των contextual εφαρμογών. Η αναγνώριση της θέσης θα ενισχύσει επίσης τα συστήματα, όπως η κινητή και η κοινωνική δικτύωση μέσω κινητού. Η αυξανόμενη της αναγνώρισης θέσης χρησιμοποιώντας Wi-Fi ανοίγει μια σειρά νέων εφαρμογών εκμετάλλευοντας την θέσης του εξοπλισμού ή των ανθρώπων.

802.11n

Το 802.11n Wi-Fi ενισχύεται σε ταχύτητες δεδομένων μεταξύ των 100 Mbps και 300 Mbps, Το 802.11n είναι πιθανό να γίνει ένα μακρά διάρκειας πρότυπο που θα καθορίσει τις Wi-Fi επιδόσεις για

μερικά χρόνια. Το υψηλής ταχύτητας Wi-Fi είναι επιθυμητό να μεταδίδει πολυμέσα γύρω από το σπίτι και το γραφείο. Από οργανωτική άποψη, 802.11n είναι ενοχλητικοί, είναι πολύπλοκο για να ρυθμιστεί και απαιτεί την εγκατάσταση πολλών νέων access points, νέα δίκτυα και νέους clients, ώστε να ανταγωνιστεί το πρότυπο Ethernet. Ωστόσο, το 802.11n είναι η πρώτη τεχνολογία Wi-Fi για να προσφέρει επιδόσεις σε ισότιμη βάση με τα 100 Mbps του Ethernet.

Display Technologies

Οι Display τεχνολογίες περιορίζουν πολλά χαρακτηριστικά τόσο κινητών συσκευών όσο και των εφαρμογών. Κατά τη διάρκεια του 2009 και 2010, πολλές νέες Display τεχνολογίες θα έχουν αντίκτυπο στην αγορά, συμπεριλαμβανομένων των active pixel displays, και των μικρών projectors. Επίσης θα γίνεται χαμηλότερη κατανάλωση μπαταρίας, ενώ η καλύτερη απεικόνιση επιτρέπει την προβολή εικόνων και πληροφοριών που θα διανέμονται πιο εύκολα. Παθητικές απεικονίσεις σε συσκευές, όπως το e-book, προσφέρουν νέους τρόπους για τη διανομή και την κατανάλωση των εγγράφων.

Mobile Web and Widgets

Το Web για κινητά, αναδύεται ως ένας χαμηλού κόστους τρόπος για την παροχή απλών κινητών εφαρμογών σε μια σειρά από συσκευές. Έχει κάποιους περιορισμούς που δεν θα αντιμετωπιστούν έως το 2010 (για παράδειγμα, δεν θα υπάρχουν παγκόσμια πρότυπα για πρόσβαση μέσω browser). Ωστόσο, το Web για κινητά προσφέρει ένα συναρπαστικό συνολικό κόστος ιδιοκτησίας ως πλεονέκτημα σε σχέση με άλλες εφαρμογές εφαρμογές. Τα widgets (μικρά applets για το Web κινητών) υποστηρίζονται από πολλούς browsers για κινητά, και παρέχουν έναν απλό τρόπο να τροφοδοτούν συσκευές και μικρές οθόνες. Κινητές εφαρμογές για το Web θα είναι ένα μέρος των πιο B2C κινητών στρατηγικών.

Cellular Broadband

Το ασύρματο ευρυζωνικό δίκτυο έκανε μεγάλες προόδους το 2008, καθοδηγούμενο από τη διαθεσιμότητα των τεχνολογιών, όπως η πρόσβαση σε υψηλής ταχύτητας πακέτα downlink και uplink, σε συνδυασμό με τις ελκυστικές τιμές από τους φορείς κινητής τηλεφωνίας. Η απόδοση της πρόσβασης σε υψηλής ταχύτητας μεταφορά πακέτων (high-speed packet access, HSPA) παρέχει ένα megabit ή δύο από το εύρος ζώνης στις uplink και downlink κατευθύνσεις. Σε πολλές περιοχές, το HSPA παρέχει επαρκή σύνδεσιμότητα για την αντικατάσταση των Wi-Fi "hot spots," και η διαθεσιμότητα των “ώριμων” chipsets επιτρέπει στους οργανισμούς να αγοράσουν φορητούς υπολογιστές με ενσωματωμένο module για κινητά το οποίο παρέχει ανώτερες επιδόσεις σε επιπρόσθετες κάρτες.

Near Field Communication (NFC)

Το NFC παρέχει έναν απλό και ασφαλή τρόπο από τις συσκευές χειρός να επικοινωνούν σε αποστάσεις ενός ή δύο εκατοστών. Το NFC αναδύεται ως ηγετικό πρότυπο για εφαρμογές όπως πληρωμές μέσω κινητών συσκευών, με επιτυχημένες δοκιμές που πραγματοποιούνται σε μερικές χώρες. Επίσης έχει ευρύτερες εφαρμογές, όπως το "άγγιγμα ανταλλαγής πληροφοριών" (για παράδειγμα, να μεταφέρει μια εικόνα από μια φορητή συσκευή σε μια ψηφιακή κορνίζα, ή για μια φορητή συσκευή για να πάρει ένα εικονικό κουπόνι έκπτωσης).

[3]

12.3 Ανακεφαλαίωση

Σε αυτό το κεφάλαιο έγινε μία μικρή αναφορά στις τεχνολογίες οι οποίες αναμένονται να κάνουν πάταγο στον χώρο των κινητών συσκευών, τα επόμενα χρόνια.

Κεφάλαιο 13

Η περιήγηση στον κόσμο της Java Micro Edition έφτασε στο τέλος της. Το ταξίδι όμως το οποίο μπορεί να κάνει κάποιος, είναι χωρίς τελικό προορισμό. Καθημερινά αναπτύσσονται όλο και περισσότερο οι κινητές συσκευές, αποκτώντας καινούριες δυνατότητες και κανούρια θέση στην ζωή μας. Με αυτή την εργασία έγινε μία μικρή προσπάθεια να εισαχθεί κάποιος στον χώρο αυτό. Ελπίζω ο αναγνώστης να θεώρησε πως άξιζε τον κόπο.

Παρακάτω ακολουθεί το Παράρτημα, με αρκετές σημαντικές πληροφορίες για κάποιον που θέλει να κατανοήσει κάποιες έννοιες και τεχνολογίες καλύτερα.

Παράρτημα Α

Π.1 Java Community Process

Εικόνα Π.1 - Το σύμβολο της Java Community Process

Η Java Community Process ή JCP, η οποια δημιουργήθηκε το 1998, είναι μια επισημοποιημένη process η οποια επιτρέπει σε ενδιαφερόμενες ομάδες, να αναμιχθούν με τον μελλοντικό ορισμό εκδόσεων και χαρακτηριστικών της Java platform.

Η JCP συνεπάγει την χρήση των Java Specification Requests (JSRs) – τα επίσημα έγγραφα τα οποία περιγραφουν προτεινόμενες προδιαγραφές και τεχνολογίες για να προστεθούν στην Java Platform. Επίσημες δημόσιες αναθεωρήσεις λαμβάνουν χωρα πριν ένα JSR γίνει τελικό και η JCP Executive Committee ψηφίσει για αυτό. Ένα τελικό JSR παρέχει μια υλοποίηση προδιαγραφής η οποια είναι δωρεάν υλοποίηση της τεχνολογίας στον πηγαίο κώδικα κι ένα Techology Compatibility Kit για να επιβεβαιώσει την προδιαγραφή του API.

Η JCP έχει αναπτύξει ένα πρόγραμμα το οποίο είναι το πιο δημοφιλές και επιτυχημένο σύνολο από πρότυπα στην βιομηχανία λογισμικού με πάνω από 300 προδιαγραφές να έχουν δημιουργηθεί. Το JCP πρόγραμμα ορίζει μία κοινή διαδικασία κι ένα framework ώστε να λειτουργήσουν όλα τα πρότυπα της Java τεχνολογίας κι έχει καθορίσει ένα ελάχιστο επίπεδο απαιτήσεων και προσδοκιών για αυτά.

[46]

Π.2 JSR

Τα Java Specification Request είναι τα διάφορα επιμέρους τμήματα τα οποία αποτελούν την γλώσσα Java στις διάφορες εκδόσεις της. Εκδίδονται από την Java Community Process, ενώ κάθε βιβλιοθήκη η οποία ανήκει στη γλώσσα, είναι μέρος κάποιου JSR [46].

Π.3 K Virtual Machine

Η K Virtual Machine (KVM) είναι ένα πολύ σημαντικό χαρακτηριστικό της J2ME αρχιτεκτονικής είναι μία συμπαγής φορητή υψηλού επιπέδου Java Virtual Machine σχεδιασμένη από το μηδέν για συσκευές με περιορισμένους πόρους μνήμης, συνδεδεμένες σε δίκτυο, όπως τα κινητά τηλέφωνα, οι

pagers και τα personal organizers. Αυτές οι συσκευές συνήθως περιέχουν 16 ή 32 bit επεξεργαστές και πολύ μικρή μνήμη γύρω στα 128 KB. Παρ'όλα αυτά, η KVM μπορεί να χρησιμοποιηθεί ευρέως σε μία μεγάλη ποικιλία από συσκευές σε διάφορους βιομηχανικούς κλάδους αλλά και όσον αφορά το μεγάλο φάσμα των αλληλεπιδράσεων μεταξύ επεξεργαστικής ισχύος, του μεγέθους μνήμης, των χαρακτηριστικών της συσκευής, και της λειτουργικότητας της εφαρμογής.

Πιο συγκεκριμένα, η KVM σχεδιάστηκε για να είναι:

Μικρή, με ελάχιστη μνήμη από 40 – 80 KB στον πυρήνα της virtual machine
"Καθαρή", με καλό σχολιασμό και με μεγάλη φορητότητα
Όσο περισσότερο ολοκληρωμένη και γρήγορη γίνεται, χωρίς να πρέπει να θυσιαστούν οι στόχοι οι οποίοι αφορούν τον σχεδιασμό

Το K στο KVM αναπαριστά το Kilo. Ονομάστηκε έτσι, επειδή η κατανάλωση μνήμης της μετράται σε KiloBytes. Η KVM είναι κατάλληλη για 16/32-bit RISC/CISC μικροεπεξεργαστές με συνολική χρήση μνήμης όχι μεγαλύτερη των 128 KB.

Σε μερικές υλοποιήσεις, η KVM χρησιμοποιείται πάνω από μία υπάρχουσα στοίβα λογισμικού, ώστε να δώσει την δυνατότητα στην συσκευή να κατεβάσει, και να τρέξει δυναμικό διαδραστικό, ασφαλές Java περιεχόμενο στην συσκευή. Σε άλλες υλοποιήσειςη KVM χρησιμοποιείται σε χαμηλότερο επίπεδο για να υλοποιήσει το χαμηλότερο επίπεδο λογισμικού συστήματος και εφαρμογών της συσκευής σε Java.

Η KVM και CLDC είναι στενά συνδεδεμένα. Το CLDC τρέχει μόνο πάνω στην KVM και το CLDC είναι το μοναδικό configuration το οποίο υποστηρίζεται από την KVM. Στο μέλλον αναμένεται ότι το CLDC για μπορεί να τρέξει και σε άλλες J2ME Vms ενώ η KVM θα υποστηρίξει και άλλα configurations.

Περιορισμοί της KVM:

- Δεν υπάρχει JNI ή reflection.
- Δεν υπάρχουν thread groups ή daemon threads
- Δεν υπάρχουν application-defined class loaders

Πέραν των ανωτέρω περιορισμών, το CLDC απαιτεί επίσης το class verification να γίνει διαφορετικά. Τα class αρχεία επεξεργάζονται ένα class verifier εκτός συσκευής, μία διαδικασία η οποία ονομάζεται preverification. Κατά το χρόνο εκτέλεσης, η VM χρησιμοποιεί πληροφορίες οι οποίες εισάγονται στα class αρχεία από τον preverifier για να εκτελέσει τα τελικά verification βήματα. Τα αρχεία τα οποία δεν έχουν υποβληθεί σε επεξεργασία από τον preverifier δεν φορτώνονται, δεδομένου ότι δεν μπορούν να γίνουν verified [7].

Υλοποίηση

Η KVM είναι υλοποιημένη στην γλώσσα C, έτσι μπορεί να μεταφερθεί εύκολα στις διάφορες πλατφόρμες στις οποίες υπάρχει ένας C compiler. Η virtual machine έχει φτιαχτεί πάνω σε έναν απλό bytecode intrpreter με διάφορα compile-time flags και επιλογές, με σκοπό να ενισχύσουν προσπάθειες μεταφερσιμότητας και βελτιστοποίησης χώρου.

Java Code Compact (ROMizer)

Η KVM υποστηρίζει το utility JavaCodeCompact. Αυτό το utility επιτρέπει στα Java classes να συνδέονται απευθείας στην virtual machine, μειώνοντας το χρόνο εκκίνησης αρκετά. Μπορεί η τεχνική αυτή να είναι λιγότερο ευέλικτη, όμως μειώνει αρκετά τις απαιτήσεις μνήμης.

Το JavaCodeCompact μπορεί να:

- Συνδυάσει πολλαπλά αρχεία εισόδου
- Να αποφασίσει το layout και το μέγεθος του instance ενός αντικειμένου.
- Να φορτώσει μόνο συγκεκριμένα class members, αφήνοντας τα άλλα.

[35]

Π.4 CVM

Η CVM είναι μία virtual machine η οποία αναποκρίνεται στις απαιτήσεις της CDC προδιαγραφής. Παρέχει όλα τα χαρακτηριστικά τα οποία απαιτούνται κι ενσωματώνει έναν garbage collector ο οποίος είναι βελτιστοποιημένος για περιβάλλοντα με μικρή μνήμη. Με σκοπό να μειώσει τον χρόνο εκκίνησης και το memory overhead, η CVM έχεις τις Java classes prelinked κατά τον χρόνο του build (κάτι το οποίο κάνει και η KVM) [37].

Π.5 Manifest file

Στην Java platform, ένα manifest αρχείο, είναι ένα συγκεκριμένο αρχείο το οποίο περιέχεται μέσα σε ένα αρχείο JAR. Χρησιμοποιείται για να ορίζει την έκταση και τα σχετικά με τα δεδομένα πακέτα. Είναι ένα meta data αρχείο το οποίο περιέχει ονόματα με κάποιες τιμές οργανωμένα σε διαφορετικούς τομείς. Εάν ένα JAR αρχείο σκοπεύεται να χρησιμοποιηθεί σαν εκτελέσιμο, τότε το manifest δείχνει την main class της εφαρμογής. Το manifest αρχείο ονομάζεται MANIFEST.MF [50].

Π.6 JAD file

Το Java Application Descriptor (JAD) είναι μία μορφή αρχείου η οποία χρησιμοποιείται για την διαχείριση και την περιγραφή των MIDlets τα οποία περιέχει ένα JAR. Κάθε Jar αρχείο πρέπει να έχει το δικό του JAD αρχείο το οποίο συμπεριλαμβάνεται σαν ένα ξεχωριστό αρχείο στο JAR πακέτο.

Το JAD περιέχει συγκεκριμένα χαρακτηριστικά πάνω στο configuration τα οποία μπορεί να χρησιμοποιηθούν όταν εγκαθίσταται ένα MIDlet. Τα JAD αρχεία, είναι αρχεία κειμένου, τα οποία μπορούν να δημιουργηθούν και να τροποποιηθούν με έναν text editor. Οι παράμετροι του JAD μπορούν να μπουν με οποιαδήποτε σειρά.

For internet distribution, the MIME-type text/vnd.sun.j2me.app-descriptor must be set for JAD files in

the web server.

Πίνακας Π.1 - JAD and MANIFEST.MF παράμετροι

Παράμετρος	JAD	Manifest	Περιγραφή
MIDlet-Name	X	X	Το όνομα του MIDlet suite το οποίο υπάρχει μέσα στο JAR αρχείο. Αυτό είναι το όνομα το οποίο φαίνεται στον χρήστη.
MIDlet-Version	X	X	Ο αριθμός έκδοσης του MIDlet. Ο αριθμός αυτός θα πρέπει να έχει τρία ψηφία στην μορφή n.n.n..
MIDlet-Vendor	X	X	Το όνομα του παρόχου του MIDlet.
MicroEdition-Profile	X*	X*	Η έκδοση των εκδόσεων της MIDP προδιαγραφής, στην οποία το MIDlet μπορεί να δουλέψει.
MicroEdition-Configuration	X*	X*	Το Java ME configuration (CLDC) το οποίο απαιτείται από τα MIDlets.
MIDlet-n	X*	X*	Χαρακτηριστικό το οποίο αναγνωρίζει το MIDlet μέσα σε ένα MIDlet suite. Όπου n ένας αριθμός >= 1. Όλα τα MIDlets θα πρέπει να προστεθούν με αυτόν τον τρόπο.
MIDlet-Jar-URL	X	-	Το URL του JAR αρχείου το οποίο περιέχει το MIDlet.
MIDlet-Jar-Size	X	-	Το αρχείο του MIDlet JAR αρχείου σε bytes.
MIDlet-Description	O	O	Μία περιγραφή του MIDlet για τον χρήστη.
MIDlet-Icon	O	O	Ένα εικονίδιο το οποίο αναπαριστά την εφαρμογή μέσα στο μενού επιλογών του κινητού.
MIDlet-Info-URL	O	O	Το URL ενός αρχείου το οποίο περιέχει περισσότερες πληροφορίες περιγράφοντας το MIDlet suite.
MIDlet-Data-Size	O	O	Το ελάχιστο ποσό χώρου αποθήκευσης, τον οποίο το MIDlet suite απαιτεί.
MIDlet-Install-Notify	O	O	Ένα URL το οποίο χρησιμοποιείται για να αναφέρει την επιτυχία ή την αποτυχία της εγκατάστασης ενός MIDlet σε έναν απομακρυσμένο server.
MIDlet-Delete-Notify	O	O	Ένα URL το οποίο χρησιμοποιείται για να αναφέρει την επιτυχία ή την αποτυχία της απεγκατάστασης ενός MIDlet σε έναν απομακρυσμένο server.
MIDlet-Permissions	O	O	Αιτήσεις άδειας από το MIDlet. Πολλαπλές άδειες μπορούν να δωθούν χωρισμένες με ένα κόμμα.
MIDlet-Permissions-Opt	O	O	Προαιρετικές αιτήσεις άδειας.

Παράμετρος	JAD	Manifest	Περιγραφή
MIDlet-Push-n	O	O	Push static registration.
MIDlet-specific attributes	O	O	Οι προγραμματιστές MIDlet μπορούν να παρέχουν περιορισμένη ρυθμισιμότητα για MIDlets συμπεριλαμβάνοντας χαρακτηριστικά τα οποία μπορούν να ανακτηθούν κατά την εκτέλεση.
MIDlet-Jar-RSA-SHA1	O	-	Ορίζει την υπογραφή του JAR (για έμπιστα MIDlets μόνο).
MIDlet-Certificate-n-m	O	-	Ορίζει το πιστοποιητικό δημοσίου κλειδιού (για έμπιστα MIDlets μόνο).

X = υποχρεωτικό

X* = πρέπει να υπάρχει είτε στο MANIFEST.MF ή στο JAD αρχείο

O = προαιρετικό

- = δεν υποστηρίζεται

Όταν τα JAD και MANIFEST.MF αρχεία έχουν χαρακτηριστικά τα οποία αντικρούονται, τότε το JAD αρχείο υπερβαίνει τα χαρακτηριστικά του MANIFEST.MF αρχείου.

Δείγμα manifest αρχείου:

```
MIDlet-Name: MyMIDlet
MIDlet-Version: 0.0.1
MIDlet-Vendor: Nokia
MicroEdition-Profile: MIDP-2.0
MicroEdition-Configuration: CLDC-1.1
MIDlet-1: HelloWorldMIDlet, , example.hello.HelloWorldMIDlet
```

Δείγμα JAD αρχείου:

```
MIDlet-Name: MyMIDlet
MIDlet-Version: 0.0.1
MIDlet-Vendor: Nokia
MicroEdition-Profile: MIDP-2.0
MicroEdition-Configuration: CLDC-1.1
MIDlet-Jar-URL: MyMIDlet.jar
MIDlet-Jar-Size: 1887
MIDlet-1: HelloWorldMIDlet, , example.hello.HelloWorldMIDlet
```

[16]

Π.7 Application management software

Το application management software είναι ένα μέρος του περιβάλλοντος του λειτουργικού συστήματος το οποιο διαχειρίζεται τα MIDlets. Διατηρεί τις καταστάσεις και καθοδηγεί τα MIDlets μέσα από τις αλλαγές των καταστάσεων αυτών.

Π.8 JNI

Το Java Natice Interface (JNI) είναι ένα προγραμματιστικό framework τα οποίο επιτρέπει στον Java κώδικα, ο οποίος τρέχει σε μία JVM, να καλεί και να καλείται από native applications (προγράμματα συγκεκριμένα προς το hardware και το Λειτουργικό Σύστημα της πλατφόρμας) και βιβλιοθήκες σε άλλες γλώσσες, όπως οι C, C++ και assembly [44].

Π.9 Java Device Test Suite

Το Java Device Test Suite (JTDS) είναι ένα βιομηχανικό πρότυπο εργαλείο για την αξιολόγηση υλοποιήσεων σε Java ME Platform. Αυτό το εργαλείο εκτελεί ποιοτικά test για συσκευές οι οποιες χρησιμοποιούν την Java ME platform.Ένα χαρακτηριστικό το οποιο διακρίνει το Java Device Test Suite από τα TCKs είναι η επικέντρωση του στην ποιότητα της υλοποίησης.

Το Java Device Test Suite είναι ένα επεκτάσιμο σύνολο πακέτων από test, καθώς και μια κοινή μονάδα διαχείρισης, τα οποία μπορούν να χρησιμοποιηθούν για να έχουν πρόσβαση στην ποιότητα κάθε σύσκεψης η οποια υλοποιεί έναν συμβατό συνδυασμό των Java ME τεχνολογιών, συμπεριλαμβανομένων των παρακάτω:

Πίνακας Π.2 - JSRs τα οποία ελέγχει το JTDS

Τεχνολογία	**Προδιαγραφή**
Personal Digital Assistant (PDA) optional packages	JSR 75
Java APIs for Object Exchange (OBEX) and Bluetooth	JSR 82
Mobile Information Device Profile (MIDP 1.0 & 2.0)	JSRs 37, 118
Mobile Media API (MMAPI 1.0 & 1.1)	JSR 135
Connected Limited Device Configuration (CLDC 1.0 & 1.1)	JSRs 30, 139
Web Services API (WSA), includes JAXP and JAX-RPC	JSR 172
Security and Trust Services API (SATSA)	JSR 177
Location API (LAPI) optional package	JSR 179
Session Initiation Protocol (SIP)	JSR 180
Mobile 3D Graphics API	JSR 184
Java Technology for the Wireless Industry	JSR 185
Wireless Messaging API (WMA 1.0 & 1.1 & 2.0)	JSRs 120, 205
Content Handler API (CHAPI)	JSR 211

Τεχνολογία	Προδιαγραφή
Scalable Vector Graphics (SVG) extension to 2D	JSR 226
Payment API (PAPI)	JSR 229
Advanced Multimedia Supplement (AMMS)	JSR 234
Internationalization (Mobile I18N)	JSR 238

Το Java Device Test Suite μπορεί να χωριστεί σε τρεις κυριες κατηγοριες:

- Benchmark test τα οποια συγκρίνουν την απόδοση μιας συσκευής με ένα reference standard
- Test ετοιμότητας, τα οποια αποκτούν πρόσβαση στην ικανότητα της συσκευής να τρέχει test και να ανακαλύπτει τα APIs τα οποια υποστηρίζει η σύσκεψη.
- Γενικά test (χωρισμένα σε πακέτα από test)

Τα test τα οποια περιέχονται σε πακέτα από test μπορούν να χωριστούν σε μερικές ομάδες by tested subsystems:

Τα test Over-the-air (OTA) επιβεβαιώνουν ότι μια σύσκεψη μπορεί να υλοποιήσει life cycle operations και μπορεί να επικοινωνήσει με έναν server ο οποίος θα τροφοδοτεί συνεχώς με πληροφορίες
Τα Test ασφαλειας επιβεβαιώνουν την σωστή υλοποίηση μοντέλων πιστοποιητικών, άδεις και πολιτικών ασφαλειας
Για Test δικτύου επιβεβαιώνουν υλοποίηση "διαφορετικών" πρωτοκόλλων όπως: HTTP, HTTPS, Socket, UDP, SMS, Bluetooth και αλλα. Μερικά σύνολα από test επιβεβαιώνουν το κανάλι μεταξύ δυο υλοποιήσεων
Για Test GUI "επαληθεουν" υλοποιήσεις γραφικού περιβάλλοντος για διαφορετικά αντικείμενα
Τα Test virtual machine επιβεβαιώνουν υλοποίηση του VM πυρήνα

Το Java Device Test Suite έχει περίπου 11.000 test τα οποια μπορούν να επεκταθούν από τον κάθε έναν. Οι χρηστες μπορούν να τρέχουν οποιοδήποτε συνδυασμό από test, ανάλογα με τα χαρακτηριστικά που υποστηρίζονται από την σύσκεψη, καθώς και από τους διαθέσιμους πόρους της.

[59]

Π.10 Emulators

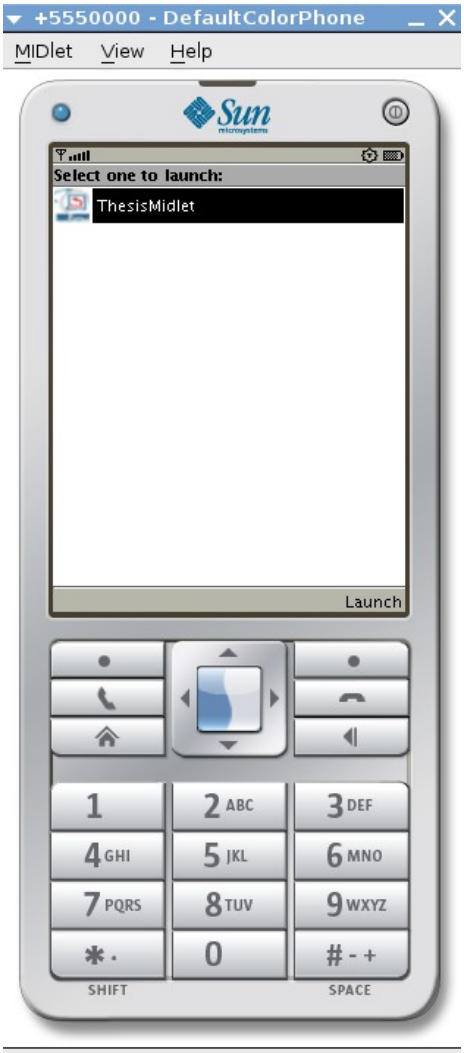
Emulator, ή όπως θα λέγαμε στα ελληνικά προσομοιωτής, για την J2ME, είναι ένα πρόγραμμα, είτε μόνο του, είτε τμήμα ενός IDE, το οποίο προσομοιώνει την λειτουργία ενός κινητού. Μπορεί να προσωμοιώσει όλες τις λειτουργίες του, ακόμα και επικονωνία μεταξύ κινητών (έχοντας δύο instances του προγράμματος ανοιχτά).

Η υπάρξει των emulators είναι πολύ αναγκαία. Η ανάπτυξη μίας εφαρμογής η οποία θα θέλει συνεχώς ελέγχους σε μία κινητή συσκευή, θα ήταν πραγματικά από πολύ δύσκολη έως ακατόρθωτη χωρίς την χρήση ενός τέτοιου προγράμματος. Ας φανταστεί κάποιος πως κάθε φορά θα πρέπει να αλλάζει τον κώδικα, να το κάνει compile, να το μεταφέρει στην κινητή συσκευή, να το κάνει εγκατάσταση αν χρειάζεται και να το τρέξει. Κι αυτό θα πρέπει να γίνεται για κάθε αλλαγή η οποία απαιτεί έλεγχο για την σωστή λειτουργία. Να αναφερθεί βεβαίως και το debugging το οποίο δεν θα γινόταν καθόλου

εύκολα σε μία εφαρμογή που χρησιμοποιεί γραφικά, όπως ένα παιχνίδι.

Πολλές φορές όμως, ακόμα και τέτοιες εφαρμογές δεν είναι ικανές να προσομοιώσουν πλήρως το παραγόμενο αποτέλεσμα, καθώς οι διαφορετικές υλοποιήσεις των JSRs της J2ME, από κάθε κατασκευαστή, μπορεί να δώσει διαφορετικό αποτέλεσμα στην ίδια ενέργεια που κάνει ο χρήστης, σε διαφορετικές συσκευές.

Emulators προσφέρουν τα IDE Netbeans και Eclipse, καθώς και το Sun Wireless Toolkit, και εταιρείες οι οποίες κατασκευάζουν κινητές συσκευές. Αρκετά από τα emulators, δεν λειτουργούν αυτόνονα, αλλά πρέπει να προστεθούν σε κάποια από τα τρία προγράμματα τα οποία αναφέρθηκαν πιο πάνω.



Εικόνα Π.2 - Sun Emulator

Παρακάτω είναι ένα link για μερικά emulators από διάφορες εταιρείες.

[25]

Π.11 Technology Compatibility Kit

Το Technology Compatibility Kit (TCK) είναι μια σουίτα από test τα οποια τουλάχιστον ονομαστικά, ελέγχουν μια συγκεκριμένη υποτιθέμενη υλοποίηση της Java Specification Request (JSR) για συμβατότητα. Είναι ένα από τα τρία απαραίτητα κομμάτια για ένα επικυρωμένο JSR κατά την Java Community Process. Τα τρία αυτά κομμάτια είναι:

- η JSR specification
- η JSR reference implementation
- το Technology Compatibility Kit (TCK)

Περιεχόμενα κι αρχιτεκτονική

Τα TCKs τείνουν να αποκτούνται από το Specification Lead ενός δοσμένου JSR. Συνήθως (αν κι όχι πάντα) αποτελούνται από μια graphical host εφαρμογεί η οποια επικοινωνεί μέσω TCP/IP με την συσκευή ή την Java Virtual Machine, στην οποια γίνεται το test. Τα Test συνήθως αποκτούνται από την σύσκεψη μέσω HTTP και τα αποτελέσματα αναρτώνται στην host application με έναν παρόμοιο τρόπο. Αυτή η αποσύνδεση ενεργοποιεί τα TCKs τα οποια θα χρησιμοποιηθούν για να "τεσταρουν" virtual machines σε "συκευες" όπως CLDC κινητά τηλεφωνα τα οποια δεν έχουν την ικανότητα να τρέχουν ολοκληρωμένη την TCK host application.

Τα test τα οποια εμπεριέχονται στο JSR υποτίθενται ότι προέρχονται από τις δηλώσεις της JSR προδιαγραφής. Οποιοδήποτε API θα έχει ένα σύνολο από test για να διασφαλίσει ότι συμπεριφέρεται με τον αναμενόμενο τρόπο, συμπεριλαμανομένου και των συνθηκών σφαλμάτων.

Με σκοπό να υπάρχει συμβατότητα με ένα JSR μια υλοποίηση Java πρέπει να περάσει το σχετικό TCK. Οποιεσδήποτε (σπάνιες) εξαιρέσεις πρέπει να διαπραγματευτούν με την ηγεσία της προδιαγραφής. Εξαιτίας αυτού, τα TCKs είναι πολύ σημαντικά στην υλοποίηση ενός JSR. Το πρώτο μεγάλο ορόσημο είναι να μπορέσει αρχικά να τρέχει το TCK, το οποίο απαραίτητα εμπεριέχει την Java υλοποίηση και το υποκείμενο δίκτυο, έχοντας ένα επίπεδο ωριμότητας.. Έπειτα το TCK θα πρέπει να ρυθμιστεί σωστά – επειδή πρέπει να είναι αρκετά ευέλικτο ώστε να μπορεί να λειτουργήσει με κάθε υλοποίηση, υπάρχουν πολλές επιλογές. Συγκεκριμένα test επίσης απαιτούν setup activity – αυτό τείνει να είναι ιδιαιτέρως πολύπλοκο για τα test τα οποια διασφαλίζουν την σωστή συμπεριφορά σε συνθήκες σφαλμάτων, επειδή η υλοποίηση Java πρέπει να τεθεί με τέτοιο τρόπο ώστε να αναπαραχθεί το σφάλμα. Τέλος, κάθε test το οποιο αποτυγχάνει θα πρέπει να διορθωθεί το οποιο συνήθως χειρίζεται από τους συνήθης μηχανισμούς εντοπισμού σφαλμάτων.
Μερικοί προγραμματιστές θεωρούν πως το προϊόν τους είναι κυρίως υλοποιημένο μόλις περάσουν τα TCKs. Ενώ αυτό είναι σωστό, ότι τα TCKs είναι αρκετά σφαιρικά, είναι πολλά σημεία όμως τα οποία δεν τα καλύπτουν. Αυτά περιλαμβάνουν θέματα όπως η απόδοση, καθώς επίσης και κάποια προαιρετικά χαρακτηριστικά. Δεν υπάρχει άλλος τρόπος, παρα να γίνουν πολλά test υπό κανονικές συνθήκες, για την αντιμετώπιση αυτών των ελλείψεων, αν και επιπρόσθετα σουίτες test όπως το JTDS θα μπορούσε να βοηθήσει.

Π.12 TCK για την Java Platform

Το Technology Compatibility Kit για μια συγκεκριμένη Java platform ονομάζεται Java Compatibility Kit (JCK). Είναι μια εκτεταμένη σουίτα test η οποια χρησιμοποιείται από την Sun Microsystems για να διασφαλίσει τις συμβατές υλοποιήσεις στην πλατφόρμα.

Π.13 TCK framework

Το εργαλείο "JavaTest" harness είναι σήμερα το πιο συχνό unit testing framework το οποο χρησιμοποείται για να επιβεβαιώσει την συμβατότητα της υλοποίησης. Είναι ένα testing framework γενικού σκοπού σχεδιασμένο να τρέχει TCK tests. Παρόλα αυτά, κάποιες προδιαγραφές χρησιμοποιούν επίσης το "Junit" [53].

Π.14 JUnit

Το Junit είναι ένα unit testing framework για την Java. Δημιουργήθηκε από τον Kent Beck και τον Erich Gamma [47].

Π.15 Preverifying

Πριν από την εκτέλεση ενός MIDlet, το αρχείο class πρέπει να γίνει pre-verified. Αυτό αποτελεί ένα επιπλέον βήμα πέρα από αυτό που απαιτείται όταν γράφεται μία παραδοσιακή εφαρμογή σε Java. Το pre-verification είναι αναγκαίο για να παρέχεται ένα ταχύτερο και πιο απλοποιημένο verified αρχείο class μόλις τα αρχεία φορτώθούν σε μία συσκευή. Όσα αρχεία class δεν γίνουν pre-verified δεν θα μπορέσουν να τρέξουν.

Π.16 Base64

Ο όρος Base64 αναφέρεται σε μια συγκεκριμένη MIME μεταφορά περιεχομένου κωδικοποίηση. Επίσης, χρησιμοποιείται ως γενικός όρος για οποιοδήποτε παρόμοιο σύστημα κωδικοποίησης το οποίο κωδικοποιεί δυαδικά δεδομένα επεξεργάζοντάς τα αριθμητικά και μετατρέποντάς τα σε μία Base64 αναπαράσταση. Η συγκεκριμένη επιλογή της βάσης οφείλεται στην ιστορία των χαρακτήρων της κωδικοποίησης: κάποιος μπορεί να επιλέξει μια σειρά από 64 χαρακτήρες οι οποίοι είναι μέρος του υποσύνολο των πιο συνηθισμένων κωδικοποιήσεων καθώς επίσης και είναι εκτυπώσιμοι. Ο συνδυασμός αυτός καθιστά τα δεδομένα απίθανο να τροποποιηθούν από την μεταφορά μεταξύ συστημάτων όπως τα emails, τα οποία δεν είναι παραδοσιακά 8-bit clean.

Το MIME Base64 χρησιμοποιεί τα γράμματα από το A-Z, a-z και τους αριθμούς από το 0-9 για τις πρώτες 62 τιμές. Υπάρχουν και άλλα παρόμοια συστήματα, που συνήθως προέρχονται από Base64, τα οποία μοιράζονται αυτή την ιδιότητα, αλλά διαφέρουν στα επελεγμένα σύμβολα τα οποία έχουν διαλεχθεί για τις δύο τελευταίες τιμές. Ένα παράδειγμα είναι το UTF-7.

Παράδειγμα

Το παρακάτω είναι ένα απόσπασμα από το βιβλίο Leviathan του Thoma Hobbes

> "Man is distinguished, not only by his reason, but by this singular passion from other animals, which is a lust of the mind, that by a perseverance of delight in the continued and indefatigable generation of knowledge, exceeds the short vehemence of any carnal pleasure."

το οποίο από μια ASCII byte ακολουθία έχει κωδικοποιηθεί σε MIME Base64:

```
TWFuIGlzIGRpc3Rpbmd1aXNoZWQsIG5vdCBvbmx5IGJ5IGhpcyByZWFzb24sIGJ1dCBieSB0aGlz
IHNpbmd1bGFyIHBhc3Npb24gZnJvbSBvdGhlciBhbmltYWxzLCB3aGljaCBpcyBhIGx1c3Qgb2Yg
dGhlIG1pbmQsIHRoYXQgYnkgYSBwZXJzZXZlcmFuY2Ugb2YgZGVsaWdodCBpbiB0aGUgY29udGlu
dWVkIGFuZCBpbmRlZmF0aWdhYmxlIGdlbmVyYXRpb24gb2Yga25vd2xlZGdlLCBleGNlZWRzIHRo
ZSBzaG9ydCB2ZWhlbWVuY2Ugb2YgYW55IGNhcm5hbCBwbGVhc3VyZS4=
```

Στο παραπάνω απόσπασμα η κωδικοποιημένη τιμή της λέξης *Man* έχει γίνει *TWFu.* Σε ASCII κωδικοποίηση τα *M, a, n* αποθηκεύονται σαν τα bytes `77`, `97`, `110`, τα οποία είναι `01001101`, `01100001`, `01101110` σε Base2. Αυτά τα τρία bytes ενώνονται μαζί σε ένα 24 bit buffer παράγοντας το `010011010110000101101110`. Ομάδες των 6 bits (6 bits έχουν ως μέγιστο 64 διαφορετικές binary τιμές) μετατρέποντας σε 4 αριθμούς (24 = 6x24) τα οποία μετατρέπονται στις αντίστοιχες Base64 τιμές.

Πίνακας Π.3 - Παράδειγμα αντικατάστασης γραμμάτων σε Base64 κωδικοποίηση

Περιχεόμενο κειμένου	**M**	**a**	**n**
ASCII	77	97	110
Bit pattern	0 1 0 0 1 1 0 1	0 1 1 0 0 0 0 1	0 1 1 0 1 1 1 0
Index	19	22	5	46
Base64 κωδικοποίηση	**T**	**W**	**F**	**u**

Όπως δείχνει το παράδειγμα, η κωδικοποίηση Base64 κωδικοποιεί 3 μη κωδικοποιημένα bytes σε 4 κωδικοποιημένους ASCII χαρακτήρες.

Το παράδειγμα το οποίο ακολουθεί δείχνει πως μειώνοντας τους χαρκατήρες πως αλλαγή επιρεάζεται το padding του παραγόμενου αποτελέσματος:

Πίνακας Π.4 - Παράδειγμα 1ο - Αντικατάσταση λέξεων σε Base64 κωδικοποίηση

Το κομμάτι εισόδου τελειώνει σε	**Το παραγόμενο αποτέλεσμα τελειώνει με**
*carnal plea**sure***	***c3VyZS4=***
*carnal plea**sure***	***c3VyZQ==***
*carnal plea**sur***	***c3Vy***
*carnal plea**su***	***c3U=***

Οι ίδιοι χαρακτήρες θα κωδικοποιηθούν διαφορετικά εξαρτώμενοι από την θέση στην οποία η ομάδα των τριών οκταδικών έχει κωδικοποιηθεί για να παράγει τους τέσσερις χαρακτήρες. Για παράδειγμα:

Πίνακας Π.5 - Παράδειγμα 2ο - Αντικατάσταση λέξεων σε Base64 κωδικοποίηση

Το κομμάτι εισόδου	Κωδικοποείται σε
*lea**sure.***	*bGVh**c3VyZS4=***
*ea**sure.***	*ZWF**zdXJlLg==***
*a**sure.***	*YX**N1cmUu***
sure.	***c3VyZS4=***

The Base-64 Alphabet

Value	Char	Value	Char	Value	Char	Value	Char
0	A	16	Q	32	g	48	w
1	B	17	R	33	h	49	x
2	C	18	S	34	i	50	y
3	D	19	T	35	j	51	z
4	E	20	U	36	k	52	0
5	F	21	V	37	l	53	1
6	G	22	W	38	m	54	2
7	H	23	X	39	n	55	3
8	I	24	Y	40	o	56	4
9	J	25	Z	41	p	57	5
10	K	26	a	42	q	58	6
11	L	27	b	43	r	59	7
12	M	28	c	44	s	60	8
13	N	29	d	45	t	61	9
14	O	30	e	46	u	62	+
15	P	31	f	47	v	63	/

(pad) =

Εικόνα Π.3 - Το Base-64 αλφάβητο

[41]

Π.17 Bluetooth

Εικόνα Π.4 - Το σήμα του Bluetooth

Το Bluetooth είναι ένα ανοικτό ασύρματο πρωτόκολο για την ανταλλαγή δεδομένων σε κοντινές αποστάσεις, από κινητές συσκευές, δημιουργόντας personal area networks (PANs). Σαν ιδέα εμφανίστηκε ως εναλλακτική στα καλώδια δεδομένων RS232. Μπορεί να συνδέσει μερικές συσκευές παρακάμπτοντας τα προβλήμα συγχρονισμού.

Υλοποίηση

Το bluetooth χρησιμοποεί τεχνολογία η οποία ονομάζεται frequency-hopping spread spectrum η οποία χωρίζει σε τμήματα τα δεδομένα τα οποία αποστέλονται και τα μεταδίδει έτσι, μέχρι και σε 79 διαφορετικές συχνότητες. Στην βασική του λειτουργία, χρησιμοποιείται το Gaussian frequency-shift keying (GFSK). Το bluetooth παρέχει έναν τρόπο για σύνδεση και ανταλλαγή πληροφοριών μεταξύ συσκευών, όπως κινητά τηλέφωνα, τηλέφωνα, laptops, υπολογιστές, εκτυπωτές, GPS, ψηφιακές κάμερες και παιχνιδομηχανές, μέσω μίας ασφαλούς μικρής εμβέλειας συχνότητας στα 2.4 Ghz. Οι προδιαγραφές του bluetooth αναπτύσσονται από το Bluetooth Special Interest Group (SIG).

Χρήση

Το bluetooth είναι ένα πρότυπο για πρωτόκολα επικοινωνίας, αρχικά σχεδιασμένο για χαμηλή κατάλωση ενέργειας, μικρής εμβέλειας, βασισμένη σε μικρού κόστους μικροκυκλώματα πομποδέκτη σε κάθε συσκευή. Το bluetooth καθιστά δυνατή την επικοινωνία μεταξύ συσκευών ενώ είναι σε απόσταση μεταξύ τους. Επειδή οι συσκευές χρησιμοποιούν ένα σύστημα ραδιοεπικοινωνίας, δεν χρειάζεται η μία να έχει οπτική επαφή με την άλλη.

Πίνακας Π.6 - Ισχύς Bluetooth κι εμβέλεια

Τάξη	Μέγιστη Επιτρεπόμενη Ισχύς mW (dBm)	Απόσταση (περίπου)
Τάξη 1	100 mW (20 dBm)	~100 μέτρα
Τάξη 2	2.5 mW (4 dBm)	~10 μέτρα
Τάξη 3	1 mW (0 dBm)	~1 μέτρο

Στις πιο πολλές περιπτώσεις η εμβέλειας τάξης 2 συσκευών μεγαλώνει αν συνδεθούν σε έναν πομποδέκτη τάξης 1. Αυτό επιτυγχάνεται χάρη στην μεγαλύτερη ευαισθησία και μετάδοση της ισχύς

συσκευών τάξης 1.

Πίνακας Π.7 - Έκδοση Bluetooth και ταχύτητα μεταφοράς δεδομένων

Έκδοση	Data Rate
Έκδοση 1.2	1 Mbit/s
Έκδοση 2.0 + EDR	3 Mbit/s
Wimedia Alliance (προτεινόμενο)	53 - 480 Mbit/s

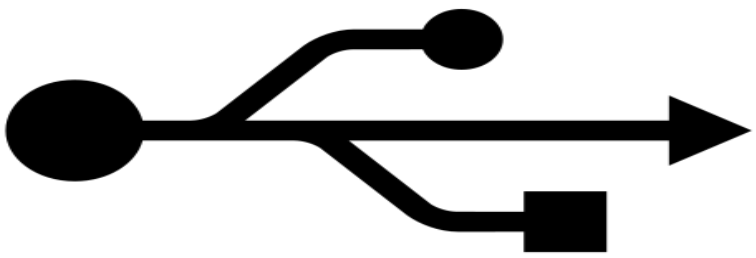
Εφαρμογές οι οποίες χρησιμοποιούν το bluetooth

Οι πιο διαδεδομένες εφαρμογές Bluetooth περιλαμβάνουν:

- Ασύρματο έλεγχος επικοινωνίας μεταξύ ενός κινητού τηλεφώνου κι ενός hands-free ακουστικού. Αυτή ήταν μία από τις πρώτες εφαρμογές που έγινε πολύ διάσημη
- Ασύσρματη επικοινωνία μεταξύ υπολογιστών σε ένα περιορισμένο χώρο όπου απαιτείται μικρό εύρος σύνδεσης
- Ασύρματη επικοινωνία μεταξύ υπολογιστή και συσκευές εισόδου εξόδου, με πιο γνωστές, το ποντίκι, το πληκτρολόγιο και τον εκτυπωτή
- Μεταφορά αρχείων, λεπτομέρειες επαφών, συναντήσεις ημερολογίου, υπενθυμίσεις μεταξύ συσκευών, με το OBEX
- Αντικατάσταση των παραδοσιακών ενσύρματων σειριακών μεθόδων επικοινωνίας σε GPS, ιατρικό εξοπλισμό, σαρωτές bar code και συσκευές ελέγχου της κυκλοφορίας
- Για ελέγχουν εκεί όπου χρησιμοποιούνταν οι υπέρηθρες
- Παιχνιδομηχανές όπως το Wii της Nintendo και το PlayStation 3 της Sony για επικοινωνία της κονσόλας με τα χειριστήρια
- Dial-up internet πρόσβαση σε υπολογιτές ή PDAs χρησιμοποιώντας ένα κινητό τηλέφωνο σαν modem

[42]

Π.18 USB



Εικόνα Π.5 - Το σήμα του USB

Το Universal Serial Bus (USB) είναι ένα πρότυπο σειριακού διαύλου για να συνδέει συσκευές σε ένα host υπολογιστή. Το USB σχεδιάστηκε για να επιτρέπει πολλές "περιφιεριακες" συσκευές να συνδεθούν σε ένα μόνο πρυτοποποιημένο intrface socket και για να βελτιώσει τις plug and play ικανότητες, επιτρέποντας hot swapping. Αυτό γίνεται επιτρέποντας σε συσκευές να συνδεθούν ή να αποσυνδεθούν χωρίς να κάνει επανεκκίνηση ο υπολογιστής ή να κλείσει το session. Αλλα βολικά χαρακτηριστικά περιλαμβάνουν την παροχή ενέργειας σε χαμηλής κατανάλωσης συσκευές, εξαλείφοντας την ανάγκη για εξωτερική παροχή ενέργειας και επιτρέπει σε πολλές συσκευές να χρησιμοποιηθούν χωρίς να απαιτεί drivers από τον κατασκευαστή για να εγκατασταθεί.

Το USB προορίζεται να αντικαταστήσει πολλές ποικιλίες σειριακών και παράλληλων θυρών. Το USB μπορεί να συνδέσει τα περιφερειακά υπολογιστών, όπως είναι τα ποντίκια, τα πληκτρολόγια, PDAs, gamepads και χειριστήρια, scanners, ψηφιακές φωτογραφικές μηχανές, εκτυπωτές, τις προσωπικές συσκευές αναπαραγωγής πολυμέσων, flash drives, και εξωτερικούς σκληρούς δίσκους. Για πολλές από αυτές τις συσκευές, USB έχει γίνει η τυποποιημένη μέθοδος σύνδεσης. Το USB έχει σχεδιαστεί για τους προσωπικούς υπολογιστές, αλλά έχει γίνει κοινό τρόπος σύνδεης και σε άλλες συσκευές όπως τα PDAs και τα βίντεο κονσόλες παιχνιδιών, όσο και ως καλώδιο ρεύματος μεταξύ μιας συσκευής και ενός μετασχηματιστή AC. Από το 2008, υπάρχουν περίπου 2 δισεκατομμύρια συσκευές USB που πωλούνται ανά έτος, και συνολικά περίπου 6 δισ. ευρώ έχουν πωληθεί μέχρι σήμερα.

Ο σχεδιασμός του USB έχει προτυποποιηθεί από το USB Implements Forum (USB-IF), έναν οργανισμό για βιομηχανικά πρότυπα ο οποιος αποτελείται από εταιρείες οι οποιες ασχολούνται με την βιομηχανία των υπολογιστών και ηλεκτρονικών. Κάποια από αυτά τα μέλη, είναι η Apple Inc., η Hewlett-Packard, η Intel, η NEC και η Microsoft.

Μεταφορά Δεδομένων

Χαμηλή ταχτύτητα: Η μεταφορά δεδομένων στο USB 1.0 είναι 1.5 Mbit/s και στόχευε σε συσκευές, όπως τα πληκτρολόγια και τα ποντίκα

Πλήρης ταχύτητα: Η μεταφορά δεδομένων στο USB 1.1 φτάνει τα 12 Mbit/s

Υψηλή ταχύτητα: Η μεταφορά δεδομένων στο USB 2.0 φτάνει τα 480 Mbit/s, ενώ όλου αυτού του τύπου USBs μπορούν να δουλέψουν και σε πλήρη ταχύτητα

Υπερταχύτητα: Η μεταφορά δεδομένων στο USB 3.0 αγγίζει τα 5 Gbit/s. Αναμένεται να εμφανιστεί στην αγορά μέχρι το 2010

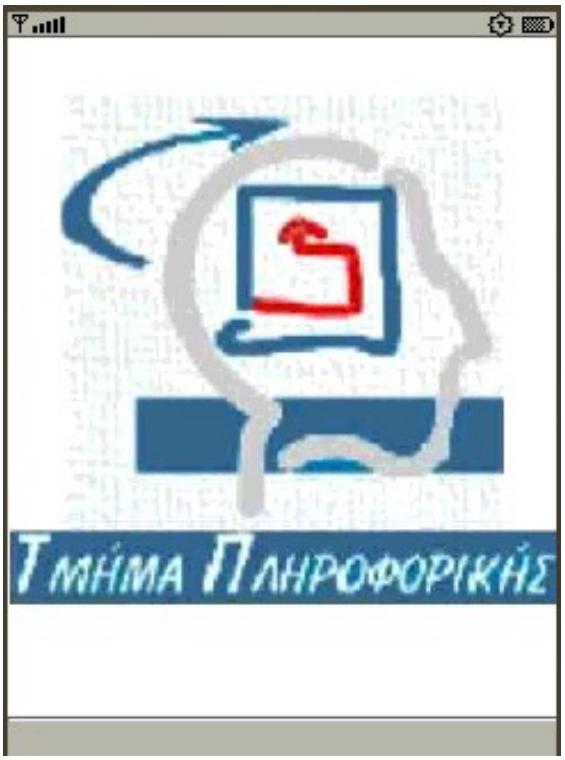
[55]

Οδηγός χρήσης εφαρμογής

Στις παρακάτω σελίδες θα γίνει μία μικρή παρουσίαση οδηγού χρήσης της εφαρμογής, ώστε ο χρήστης να εξοικιωθεί αμέσως με το περιβάλλον της.

Έναρξη προγράμματος

Με το που ξεκινήσει η εφαρμογή θα εμφανιστεί μία splash screen, η οποία μετά από 5 δευτερόλεπτα, ή από πάτημα κάποιου πλήκτρου φεύγει. Αμέσως μετά εμφανίζεται μία οθόνη από την οποία ο χρήστης μπορεί να επιλέξει την γλώσσα της αρεσκείας του, ή να τερματίσει την εφαρμογή.



Εικόνα ΟΧΕ.1 - Splash Screen

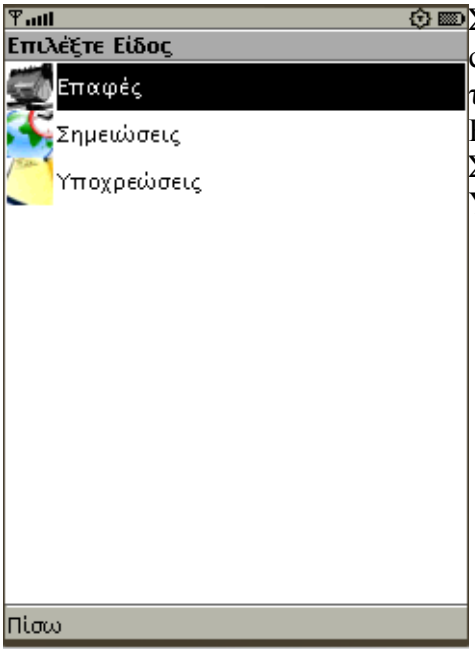
Εικόνα ΟΧΕ.2 - Επιλογής γλώσσας

Επιλογή Δράσης

Έχοντας επιλέξει την γλώσσα, ο χρήστης στην συνέχεια πρέπει να επιλέξει τι θα κάνει μέσα στο πρόγραμμα. Μπορεί να συνεχίσει στην χρήση του, ή να επιστρέψει στην αρχική οθόνη για να ξαναδιαλέξει γλώσσα. Ακόμα μπορεί να δει χρήσιμες πληροφορίες για τον τρόπο χρήσης του προγράμματος, ή κάποιες πληροφορίες σχετικές με αυτό.

Εικόνα ΟΧΕ.3 - Επιλογή δράσης

Επιλογή PIM component



Σε αυτό το σημείο ο χρήστης μπορεί να επιλέξει με ποιο component του PIM θέλει να ασχοληθεί. Οι επιλογές του είναι τρεις:
Επαφές
Σημειώσεις
Υποχρεώσεις

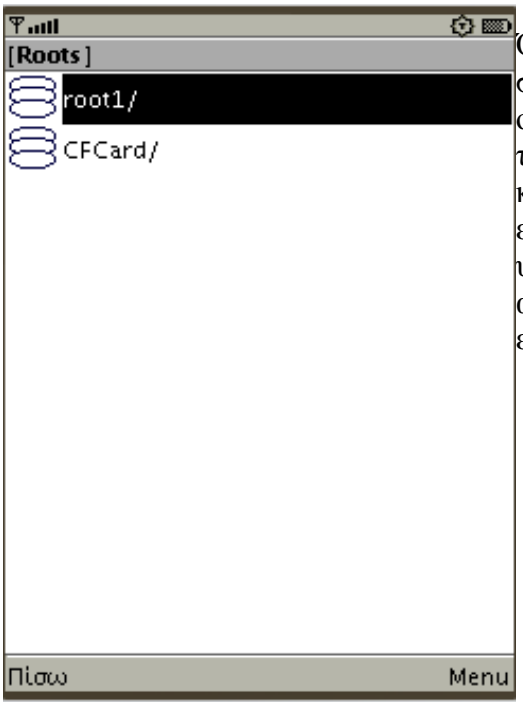
Εικόνα ΟΧΕ.4 - Επιλογή PIM component

Επιλογή Λειτουργίας

Ο χρήστης από εδώ μπορεί να επιλέξει αν θέλει να δημιουργήσει αντίγραφα ασφαλείας ή αν θέλει να επαναφέρει δεδομένα προς την κινητή του συσκευή.

Εικόνα ΟΧΕ.5 - Επιλογές για backup ή restore

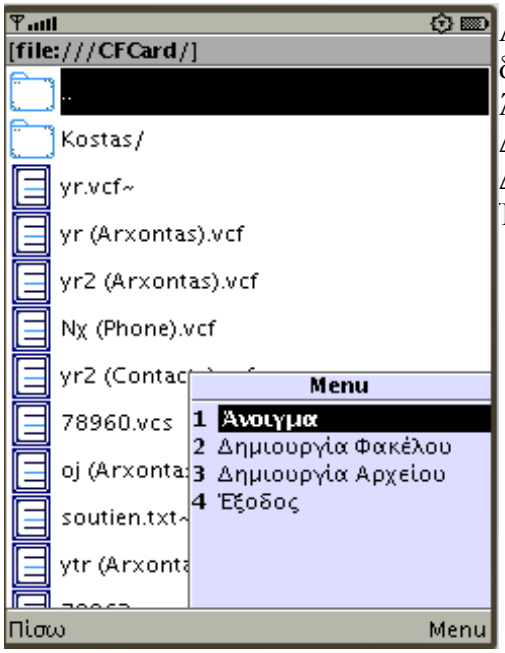
Πλοήγηση στο File System



Όποια επιλογή κι αν κάνει ο χρήστης, θα πρέπει να πλοηγηθεί στο file system της κινητής συσκευής. Το πρώτο πράγμα το οποίο θα συναντήσει είναι η απεικόνιση των roots. Δηλαδή τους τομείς αποθήκευσης. Όπως και στους υπολογιστές, έτσι κι εδώ ως ***C:*** αναπαριστάται η κύρια μνήμη της συσκευής, ενώ με το γράμμα ***E:*** η εξωτερική, σε περίπτωση που αυτή υπάρχει. Επιλέγοντας την μνήμη μπορεί ο χρήστης να δει τα αρχεία και τους φακέλους τους οποίους υπάρχουν μέσα. Στην εικόνα ΟΧΕ.6 φαίνονται τα roots "*root1"* και "*CFCard*".

Εικόνα ΟΧΕ.6 - Roots της κινητής συσκευής

Backup



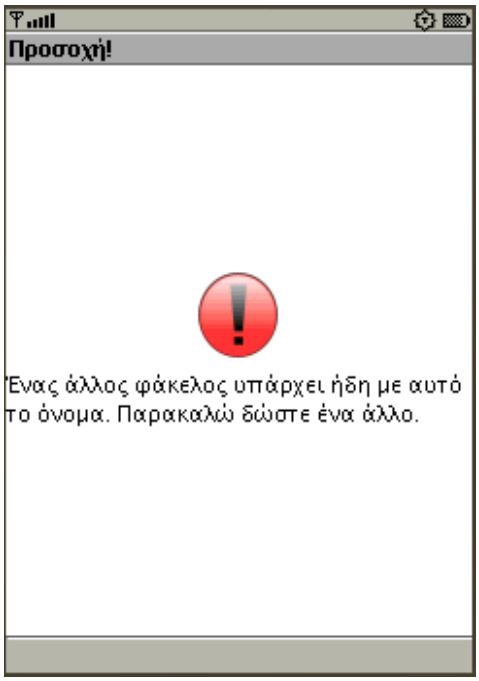
Αν επιλέξει να δημιουργήσει αντίγραφα ασφαλείας, θα του δωθούν οι παρακάτω επιλογές:
Άνοιγμα ενός φακέλου
Δημιουργία φακέλου
Δημιουργία αρχείου
Έξοδος από το πρόγραμμα

Εικόνα ΟΧΕ.7 - Επιλογές χρήστη όταν θέλει να κάνει Backup

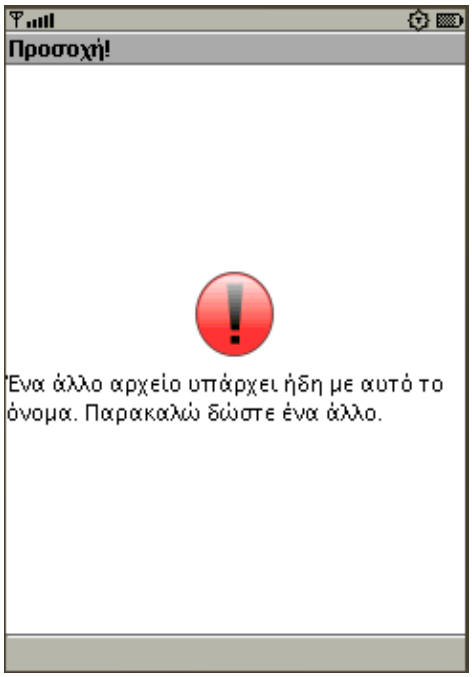
Επιλέγοντας την δημιουργία φακέλου μπορεί κάποιος να δημιουργήσει έναν ξεχωριστό φάκελο, μέσα στον οποίο ύστερα θα μπορέσει να δημιουργήσει τα αρχεία με τα δεδομένα τα οποία θέλει να σώσει. Σε κάθε περίπτωση πάντως, θα εμφανιστεί μία καινούρια οθόνη, η οποία θα ζητάει από τον χρήστη να δώσει κάποιο όνομα.

Εικόνα ΟΧΕ.8 - Ονομασία καινούριου φακέλου **Εικόνα ΟΧΕ.9 - Ονομασία καινούριου αρχείου**

Σε περίπτωση που ο χρήστης δώσει είτε ονόμα φακέλου είτε αρχείου το οποίο υπάρχει ήδη, τότε εμφανίζεται μία καινούρια οθόνη η οποία ενημερώνει τον χρήστη.



Εικόνα ΟΧΕ.10 - Μήνυμα προειδοποίησης για ήδη υπάρχων φάκελο με ίδιο όνομα



Εικόνα ΟΧΕ.11 - Μήνυμα προειδοποίησης για ήδη υπάρχον αρχείο με ίδιο όνομα

Τρόπος ονομασίας αρχείων

Όταν δημιουργηθεί το αρχείο με τα δεδομένα, αυτό θα έχει σαν όνομα το όνομα τα οποίο έδωσε ο χρήστης, ακολουθούμενο από έναν αύξοντα αριθμό και σε παρένθεση την λίστα των δεδομένων από την οποία το πήρε και την κατάληξη του αρχείου αυτού. Σε περίπτωση που το αρχείο αυτό αναφέρεται στις Επαφές, τότε δεν θα υπάρχει αύξων αριθμός, ενώ αν είναι Υποχρεώσεις, τότε πριν την κατάληξη υπάρχει το χαρακτηριστικό "*TD*" (ToDo).

Επαφές

Αν θέλουμε να σώσουμε τις Επαφές και ο χρήστης δώσει σαν όνομα την λέξη "Contacts",
τότε το αρχείο που θα δημιουργηθεί για τις επαφές της συσκευής, είναι " *Contacts (Phone).vcf*"
Ενώ αν είναι στην SIM, τότε το αρχείο θα ονομάζεται "*Contacts (SIM).vcf*".

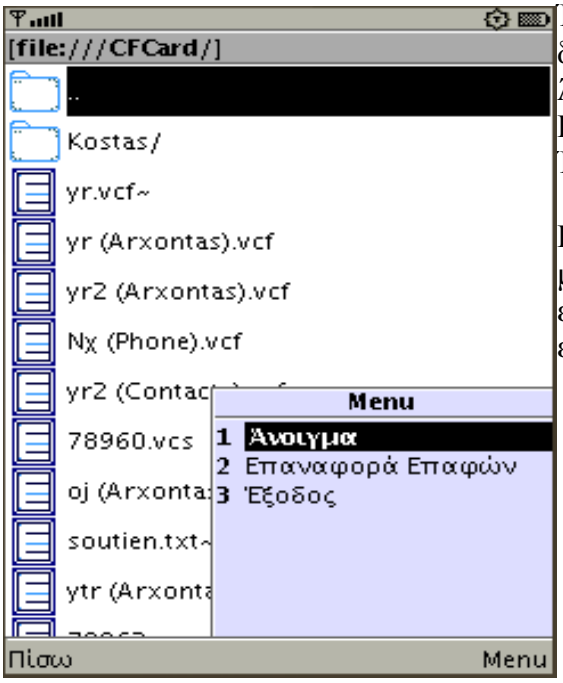
Σημειώσεις

Αν θέλουμε να αποθηκεύσουμε τα δεδομένα των Σημειώσεων και ο χρήστης δώσει σαν όνομα την λέξη "Events", τότε σε περίπτωση που έχουμε δύο υπενθυμίσεις, τα αρχεία θα ονομαστούν "*Events0 (Reminder).vcs*" και "*Events1 (Reminder).vcs*" .

Υποχρεώσεις

Αν θέλουμε να αποθηκεύσουμε τα δεδομένα των Υποχρεώσεων και ο χρήστης δώσει σαν όνομα την λέξη "ToDos", τότε σε περίπτωση που έχουμε δύο υποχρεώσεις, τα αρχεία θα ονομαστούν "*ToDos0TD.vcs*" και "*ToDos1TD.vcs*"

Restore



Έχοντας ο χρήστης επιλέξει την επαναφορά των δεδομένων, του δίνονται οι εξής επιλογές:
Άνοιγμα ενός φακέλου
Επαναφορά
Έξοδος από το πρόγραμμα

Ενώ στην οθόνη, εμφανίζονται όλα τα αρχεία τα οποία μπορεί να περιέχονται, στην επαναφορά, γίνεται επαναφορά, μόνο τα αρχεία τα οποία αφορά κάθε φορά, η επιλογή του χρήστη.

Εικόνα ΟΧΕ.12 - Επιλογές χρήστη όταν θέλει να κάνει Restore

Βιβλιογραφία

1. CollabNet Inc
   https://meapplicationdevelopers.dev.java.net/fragmentation.html

2. De Luca Stéphane 2006 – 2009
   http://mobilezoo.biz/j2me.php

3. Gartner Inc
   http://www.gartner.com/it/page.jsp?id=867012

4. Giguère Eric September 2003
   http://developers.sun.com/mobility/apis/ttips/pim/

5. Giguère Eric September 2003
   http://developers.sun.com/mobility/apis/ttips/fileconnection/

6. Giguère Eric December 2002
   http://developers.sun.com/mobility/midp/articles/optional/

7. Giguère Eric
   http://www.developer.com/java/j2me/article.php/1436051

8. Giguère Eric
   http://www.developer.com/java/j2me/article.php/1443961

9. Katz Gary May 20, 2004
   http://wireless.sys-con.com/node/44932

10. LG
    http://www.lgmobilephones.com/outlook.aspx

11. Marejka Richard
    http://java.sun.com/developer/technicalArticles/javame/msa2-intro/index.html

12. Motorola
    http://www.store.motorola.com/mot/en/US/adirect/motorola;jsessionid=9B973CD703F0E1EE6711E8AC23FDAD6E.mot1?cmd=catProductDetail&entryPoint=adirect&productID=82001089002&messageType=catProductDetail&showAddButton=true

13. Motorola
    http://www.motorola.com/consumers/v/index.jsp?vgnextoid=bda09ec8009a0210VgnVCM1000008806b00aRCRD

14. Nokia
http://europe.nokia.com/get-support-and-software/download-software/nokia-pc-suites/compatibility-and-download#74

15. Nokia forums
http://wiki.forum.nokia.com/index.php/CLDC

16. Nokia forums
http://www.forum.nokia.com/document/Java_Developers_Library_v2/?content=GUID-6858B457-7CBD-4F8F-9858-C1611D934B70.html#GUID-6858B457-7CBD-4F8F-9858-C1611D934B70

17. Nokia forums
http://www.forum.nokia.com/document/Java_Developers_Library_v2/?content=GUID-207D7DA7-A460-4E9F-AAAF-D4C96305B03A.html#GUID-207D7DA7-A460-4E9F-AAAF-D4C96305B03A

18. Nokia forums
http://www.forum.nokia.com/document/Java_Developers_Library_v2/?content=GUID-EDD762BF-BE0F-4BC8-9703-3674609C9F2D.html

19. Ortiz Enrique, August 2003
http://developers.sun.com/mobility/midp/articles/genericframework/

20. Ortiz Enrique, August 2006
http://developers.sun.com/mobility/midp/articles/msaintro/index.html

21. Ortiz Enrique, November 2007
http://developers.sun.com/mobility/getstart/articles/survey/

22. Qusay Mahmoud, December 2004
http://developers.sun.com/mobility/apis/articles/fileconnection/index.html

23. Ruuskanen Juha-pekka
http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.22.9986

24. Sing Li , Jonathan Knudsen
Beginning J2ME: From Novice to Professional, Third Edition, April 2005

25. ShareMe Technologies and Fred Grott
http://www.jroller.com/shareme/page/J2MEEmulators

26. Sony Ericsson
http://www.sonyericsson.com/cws/support/softwaredownloads/detailed/pcsuite/k550im

27. Sony Ericsson

http://www.sonyericsson.com/cws/support/softwaredownloads/detailed/sonyericssonisync/k550im

28. Sun MicroSystems
http://java.sun.com/javame/technology/msa/index.jsp

29. Sun MicroSystems
msa_datasheet.pdf

30. Sun MicroSystems
http://java.sun.com/products/jtwi/overview.html

31. Sun MicroSystems
http://java.sun.com/products/cldc/overview.html

32. Sun MicroSystems
http://developers.sun.com/mobility/apis/

33. Sun MicroSystems
http://java.sun.com/products/midp/overview.html

34. Sun MicroSystems
http://java.sun.com/javame/technology/index.jsp

35. Sun MicroSystems
J2ME Building Blocks for Mobile Devices, May 19, 2000 (KVMwp.pdf)

36. Sun MicroSystems
Java Micro Edition Connected Device Configuration (j2me_cdc.pdf)

37. Topley Kim
J2ME in a Nutshell,

38. Wells Martin J.
J2ME Game Programming, 2004

39. www.wikibooks.org
http://en.wikibooks.org/wiki/J2ME_Programming/The_J2ME_Platform

40. www.wikibooks.org
http://en.wikibooks.org/wiki/J2ME_Programming/The_J2ME_Platform

41. www.wikipedia.org
http://en.wikipedia.org/wiki/Base64

42. www.wikipedia.org
http://en.wikipedia.org/wiki/Bluetooth

43. www.wikipedia.org
http://en.wikipedia.org/wiki/ICalendar

44. www.wikipedia.org
http://en.wikipedia.org/wiki/Java_Native_Interface

45. www.wikipedia.org
http://en.wikipedia.org/wiki/JDTS

46. www.wikipedia.org
http://en.wikipedia.org/wiki/Java_Community_Process

47. www.wikipedia.org
http://en.wikipedia.org/wiki/JUnit

48. www.wikipedia.org
http://en.wikipedia.org/wiki/J2me

49. www.wikipedia.org
http://en.wikipedia.org/wiki/LCDUI#javax.microedition.lcdui

50. www.wikipedia.org
http://en.wikipedia.org/wiki/Manifest_file

51. www.wikipedia.org
http://en.wikipedia.org/wiki/MIDP

52. www.wikipedia.org
http://en.wikipedia.org/wiki/Netbeans

53. www.wikipedia.org
http://en.wikipedia.org/wiki/Technology_Compatibility_Kit

54. www.wikipedia.org
http://en.wikipedia.org/wiki/VCard

55. www.wikipedia.org
http://en.wikipedia.org/wiki/Usb

56. 版权所有
http://images.google.gr/imgres?imgurl=http://book.javanb.com/Programming-Wireless-Devices-with-the-Java2-Platform/FILES/02fig03.gif&imgrefurl=http://book.javanb.com/Programming-Wireless-Devices-with-the-Java2-Platform/0321197984_ch02lev1sec3.html&usg=__oMFeVi6Zu901RDTd6gN3enATg6w=&h=

315&w=350&sz=48&hl=el&start=85&tbnid=MSf8JH3JJ_OWOM:&tbnh=108&tbnw=120&prev=/images%3Fq%3DJ2ME%2Bwireless%2Bmarket%26gbv%3D2%26ndsp%3D20%26hl%3Del%26sa%3DN%26start%3D80

57. Nokia forums
http://www.forum.nokia.com/info/sw.nokia.com/id/1e2a0d75-526d-4715-a3a8-867eda1787a2/jsr-238-spec-fr-1_0.zip.html

58. Nokia forums
http://www.forum.nokia.com/info/sw.nokia.com/id/1e2a0d75-526d-4715-a3a8-867eda1787a2/jsr-238-spec-fr-1_0.zip.html

59. www.wikipedia.org
http://en.wikipedia.org/wiki/JDTS